IVAN VALEIJE IDOETA
FRANCISCO GABRIEL CAPUANO

40ª EDIÇÃO

ELEMENTOS DE ELETRÔNICA DIGITAL

Ivan Valeije Idoeta
Francisco Gabriel Capuano

Elementos de Eletrônica Digital

40ª Edição

São Paulo
2008 - Editora Érica Ltda.

SUMÁRIO

CAPÍTULO 1

Sistemas de Numeração

1.1 Introdução

O homem, através dos tempos, sentiu a necessidade da utilização de sistemas numéricos.

Existem vários sistemas numéricos, dentre os quais se destacam: o **sistema** decimal, o **binário**, o **octal** e o **hexadecimal.**

O sistema decimal é utilizado por nós no dia-a-dia e é, sem dúvida, o mais importante dos sistemas numéricos. Trata-se de um sistema que possui dez algarismos, com os quais podemos formar qualquer número através da lei de formação.

Os outros sistemas, em especial o binário e o hexadecimal, são muito importantes nas áreas de técnicas digitais e informática. No decorrer do estudo, perceber-se-á a ligação existente entre circuitos lógicos e estes sistemas de numeração.

1.2 O Sistema Binário de Numeração

No sistema binário de numeração, existem apenas 2 algarismos:

⇨ o algarismo **0** (zero) e

⇨ o algarismo **1** (um).

Para representarmos a quantidade zero, utilizamos o algarismo **0**, para representarmos a quantidade um utilizamos o algarismo **1**. E para representarmos a quantidade dois, se nós não possuímos o algarismo **2** nesse sistema?

É simples. No sistema decimal, nós não possuímos o algarismo dez e representamos a quantidade de uma dezena utilizando o algarismo **1** seguido do algarismo **0**. Neste caso, o algarismo **1** significa que temos um grupo de uma dezena e o algarismo **0** nenhuma unidade, o que significa dez.

No sistema binário, agimos da mesma forma. Para representarmos a quantidade dois, utilizamos o algarismo **1** seguindo do algarismo **0**. O algarismo **1** significará que temos um grupo de dois elementos e o **0** um grupo de nenhuma unidade, representando assim o número dois.

Utilizando a mesma regra, podemos representar outras quantidades, formando assim o sistema numérico. A tabela 1.1 mostra a seqüência de numeração do sistema binário até a quantidade nove.

DECIMAL	BINÁRIO
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

Tabela 1.1

Na prática, cada dígito binário recebe a denominação de **bit** (**bi**nary digi**t**), o conjunto de 4 bits é denominado **nibble** e o de 8 bits de **byte**, termo bastante utilizado principalmente na área de informática.

1.2.1 Conversão do Sistema Binário para o Sistema Decimal

Para explicar a conversão vamos utilizar um número decimal qualquer, por exemplo, o número 594. Este número significa:

$5 \times 100 + 9 \times 10 + 4 \times 1 = 594$

↓ ↓ ↓

centena dezena unidade

↑ ↑ ↑

$5 \times 10^2 + 9 \times 10^1 + 4 \times 10^0 = 594$

Esquematicamente, temos:

100	10	1
5	9	4

→ $5 \times 100 + 9 \times 10 + 4 \times 1 = 594$

10^2	10^1	10^0
5	9	4

→ $5 \times 10^2 + 9 \times 10^1 + 4 \times 10^0 = 594$

Neste exemplo, podemos notar que o algarismo menos significativo (4) multiplica a unidade (1 ou 10^0), o segundo algarismo (9) multiplica a dezena (10 ou 10^1) e o mais significativo (5) multiplica a centena (100 ou 10^2). A soma desses resultados irá representar o número.

Podemos notar ainda, que de maneira geral, a regra básica de formação de um número consiste no somatório de cada algarismo correspondente multiplicado pela base (no exemplo, o número dez) elevada por um índice conforme o posicionamento do algarismo no número.

Vamos agora utilizar um número binário qualquer, por exemplo, o número 101. Pela tabela 1.1 notamos que este equivale ao número 5 no sistema decimal.

Utilizando o conceito básico de formação de um número, podemos obter a mesma equivalência, convertendo assim o número para o sistema decimal:

2^2	2^1	2^0
1	0	1

$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

$1 \times 4 + 0 \times 2 + 1 \times 1 = 5$

O número 101 na base 2 é igual ao número 5 na base 10.

Daqui por diante, para melhor identificação do número, colocaremos como índice a base do sistema ao qual o número pertence. Assim sendo, para o exemplo podemos escrever: $5_{10} = 101_2$.

Vamos agora, fazer a conversão do número 1001_2 para o sistema decimal. Utilizando o mesmo processo, temos:

2^3	2^2	2^1	2^0
1	0	0	1

$1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 =$

$1 \times 8 + 1 \times 1 = 9_{10} \therefore 1001_2 = 9_{10}$

1.2.1.1 Exercícios Resolvidos

1 - Converta o número 01110_2 em decimal.

Primeiramente, devemos lembrar que o zero à esquerda de um número é um algarismo não significativo, logo $01110_2 = 1110_2$.

Esquematizando, temos:

2^3	2^2	2^1	2^0
1	1	1	0

$1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 =$

$8 + 4 + 2 + 0 = 14_{10} \therefore 1110_2 = 14_{10}$

2 - Converta o número 1010_2 para o sistema decimal.

2^3	2^2	2^1	2^0
1	0	1	0

$1 \times 2^3 + 1 \times 2^1 = 10_{10} \therefore 1010_2 = 10_{10}$

3 - Idem para o número 1100110001_2.

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	0	1	1	0	0	0	1

$1 \times 2^9 + 1 \times 2^8 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^0 =$

$1 \times 512 + 1 \times 256 + 1 \times 32 + 1 \times 16 + 1 \times 1 = 817_{10}$

$\therefore\ 1100110001_2 = 817_{10}$

1.2.2 Conversão do Sistema Decimal para o Sistema Binário

Como vimos, a necessidade da conversão sistema binário para decimal é evidente, pois, se tivemos um número grande no sistema binário, fica difícil perceber a quantidade que este representa. Transformando-se este número para decimal, o problema desaparece.

Veremos agora a transformação inversa, ou seja, a conversão de um número do sistema decimal para o sistema binário.

Para demonstrar o processo, vamos utilizar um número decimal qualquer, por exemplo o número 47.

Dividido o número 47 por 2, temos:

47	2
07	23

1º resto ← 1

ou seja: $2 \times 23 + 1 = 47$

ou ainda: $23 \times 2^1 + 1 \times 2^0 = 47 \rightarrow$ expressão A

Dividindo agora 23 por 2, temos:

$$23 \;\underline{|\;\; 2 \quad}$$

2º resto ← 1 11

ou seja: $11 \times 2 + 1 = 23$ → expressão B

substituindo a expressão B em A, temos:

$(2 \times 11 + 1) \times 2^1 + 1 \times 2^0 = 47$

$11 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 47$ → expressão C

Dividindo agora 11 por 2, temos:

$$11 \;\underline{|\;\; 2 \quad}$$

3º resto ← 1 5

ou seja: $5 \times 2 + 1 = 11$ → expressão D

substituindo a expressão D em C, temos:

$(2 \times 5 + 1) \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 47$

$5 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 47$ → expressão E

Dividindo 5 por 2, temos:

$$5 \;\underline{|\;\; 2 \quad}$$

4º resto ← 1 2

ou seja: $2 \times 2 + 1 = 5$ → expressão F

substituindo a expressão F em E, Temos:

$(2 \times 2+1) \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 47$

$2 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$ → expressão G

Dividindo, agora 2 por 2 temos:

$$2 \;\underline{|\;\; 2 \quad}$$

5º resto ← 0 1

último quociente ←

ou seja: $2 \times 1 + 0 = 2$ → expressão H

substituindo a expressão H em G, temos:

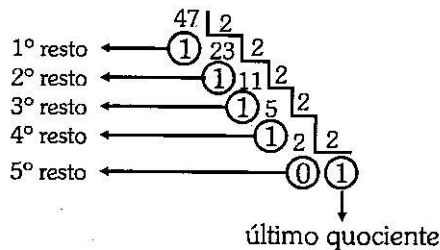
$(1x2+0) \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 47$

$1 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 47$

Esquematizando a última expressão, temos:

2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	1	1	1

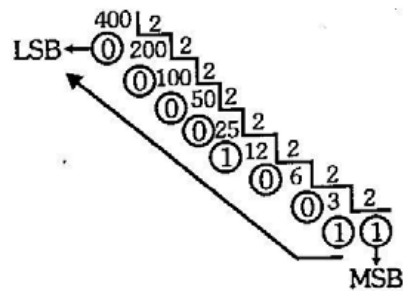
$\therefore\ 101111_2 = 47_{10}$

O processo mostra claramente a conversão e pode ser aplicado de uma forma mais simplificada, sendo denominado de método das divisões sucessivas, que consiste em efetuar-se sucessivas divisões pela base a ser convertida (no caso o 2) até o último quociente possível. O número transformado será composto por este último quociente (algarismo mais significativo) e, todos os restos, na ordem inversa às divisões. Dessa forma, temos:

O último quociente será algarismo mais significativo e ficará colocado à esquerda. Os outros algarismos seguem-se na ordem até o 1° resto:

1	0	1	1	1	1
↑	↑	↑	↑	↑	↑
último quociente	5° resto	4° resto	3° resto	2° resto	1° resto

$\therefore\ 101111_2 = 47_{10}$

Na prática, o bit menos significativo de um número binário recebe a notação de **LSB** (em inglês: **Least Significant Bit**) e o bit mais significativo de **MSB (Most Significant Bit)**.

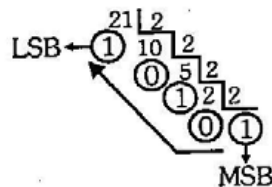
Como outro exemplo, vamos transformar o número 400_{10} em binário. Pelo método prático, temos:

Assim sendo, podemos escrever: $110010000_2 = 400_{10}$

De posse do resultado, pode-se efetuar a conversão inversa, ou seja, do sistema binário para o decimal para conferir se a operação foi efetuada corretamente.

1.2.2.1 Exercícios Resolvidos

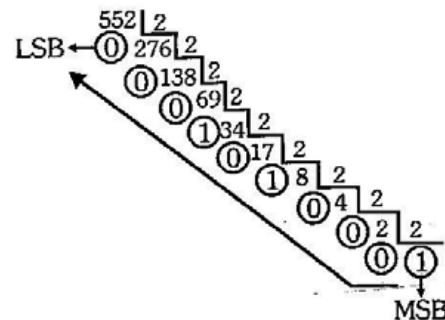
1 - Converta o número 21_{10} em binário.

Vamos utilizar o método das divisões sucessivas:

LSB ← 21 ÷ 2 … MSB

$\therefore 21_{10} = 10101_2$

Verificação: $1 \times 2^4 + 1 \times 2^2 + 1 \times 2^0 = 21$

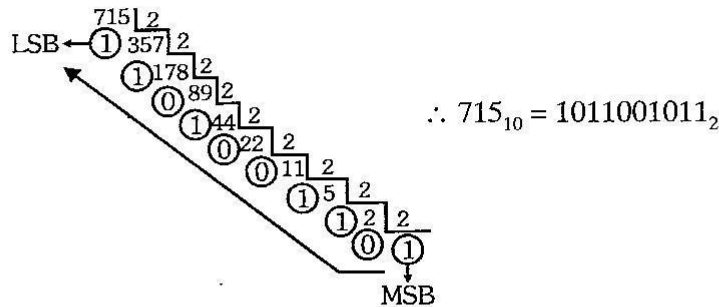
2 - Converta o número 552_{10} em binário.

Método das divisões sucessivas:

$\therefore 1000101000_2 = 552_{10}$

Verificação: $2^9 + 2^5 + 2^3 = 512 + 32 + 8 = 552_{10}$

3 - Converta o número 715_{10} em binário.

Idem aos anteriores:

$\therefore\ 715_{10} = 1011001011_2$

Verificação: $2^9 + 2^7 + 2^6 + 2^3 + 2^1 + 2^0 =$

$512 + 128 + 64 + 8 + 2 + 1 = 715_{10}$

1.2.3 Conversão de Números Binários Fracionários em Decimais

Até agora, tratamos de números inteiros. E se aparecesse um número binário fracionário? Como procederíamos para saber a quantidade que ele representa?

Para responder isso, vamos recordar primeiramente como se procede no sistema decimal. Utilizaremos, então, um número decimal fracionário qualquer, por exemplo o número 10,5. Aplicando a regra básica de formação de um número, verificamos o que ele significa:

10^1	10^0	10^{-1}
1	0	5

da tabela resulta: $1 \times 10^1 + 0 \times 10^0 + 5 \times 10^{-1} = 10{,}5$

Para números binários agimos da mesma forma. Para exemplificar vamos transformar em decimal o número $101{,}101_2$:

2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}
1	0	1	1	0	1

podemos escrever:

$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$

$$= 1 \times 4 + 0 \times 2 + 1 \times 1 + 1 \times \frac{1}{2} + 0 \times \frac{1}{4} + 1 \times \frac{1}{8} =$$

$4 + 1 + 0,5 + 0,125 = 5,625_{10}$

$\therefore 101,101_2 = 5,625_{10}$

Vamos utilizar agora, um outro número binário qualquer, por exemplo, o número 1010, 1101_2. Vamos verificar o seu valor em decimal:

2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
1	0	1	0	1	1	0	1

$1 \times 2^3 + 1 \times 2^1 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-4} =$

$$1 \times 8 + 1 \times 2 + 1 \times \frac{1}{2} + 1 \times \frac{1}{4} + 1 \times \frac{1}{16} =$$

$8 + 2 + 0,5 + 0,25 + 0,0625 = 10,8125_{10}$

$\therefore 1010,1101_2 = 10,8125_{10}$

1.2.3.1 Exercícios Resolvidos

1 - Converta o número binário $111,001_2$ em decimal.

2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}
1	1	1	0	0	1

$1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3} =$

$4 + 2 + 1 + 0,125 = 7,125_{10} \therefore 111,001_2 = 7,125_{10}$

2 - Converta o número $100,11001_2$ em decimal.

2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
1	0	0	1	1	0	0	1

$1 \times 2^2 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-5} =$

$4 + 0,5 + 0,25 + 0,03125 = 4,78125_{10}$

$\therefore 100,11001_2 = 4,78125_{10}$

1.2.4 Conversão de Números Decimais Fracionários em Binários

Podemos também converter números decimais fracionários em binários, para isso, vamos utilizar uma regra prática.

Como exemplo, vamos transformar o número 8,375 em binário. Este número significa: 8 + 0,375 = 8,375.

Vamos transformar primeiramente a parte inteira do número, como já explicado anteriormente:

LSB ← (0) — 8 ÷ 2 → 4 ÷ 2 → (0) 2 ÷ 2 → (0) (1) → MSB

$\therefore 8_{10} = 1000_2$

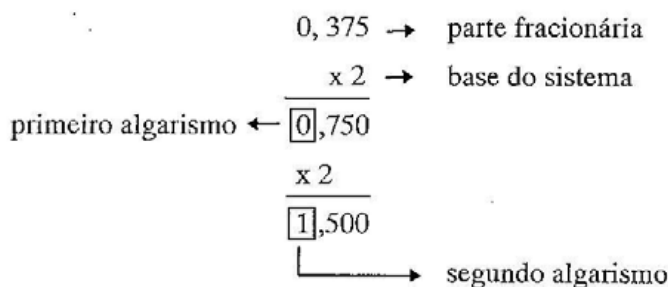
O passo seguinte é transformar a parte fracionária. Para tal, utilizaremos a regra que consiste na multiplicação sucessiva das partes fracionárias resultantes pela base, até atingir zero. O número fracionário convertido será composto pelos algarismos inteiros resultantes tomados na ordem das multiplicações. Temos, então:

0, 375 → parte fracionária

x 2 → base do sistema

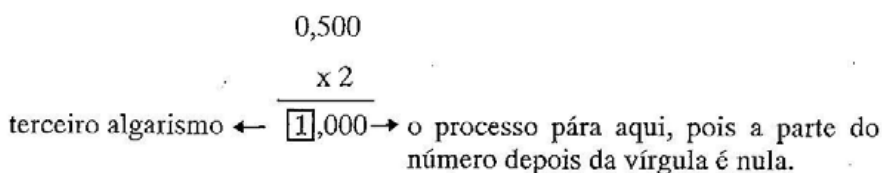
primeiro algarismo ← [0],750

x 2

[1],500 → segundo algarismo

Quando atingirmos o número 1, e a parte do número após a vírgula não for nula, separamos esta última e reiniciamos o processo:

0,500

x 2

terceiro algarismo ← [1],000 → o processo pára aqui, pois a parte do número depois da vírgula é nula.

Assim sendo, podemos escrever: $0,011_2 = 0,375_{10}$

Para completarmos a conversão, efetuamos a composição da parte inteira com a fracionária:

$1000,011_2 \therefore 8,375_{10} = 1000,011_2$

Vamos agora, transformar um outro número decimal em binário, por exemplo, o número $4{,}8_{10}$.

O primeiro passo é transformar a parte inteira do número: $4_{10} = 100_2$.

O próximo passo é converter a parte fracionária utilizando a regra já explicada:

$$\begin{array}{r} 0{,}8 \\ \times\ 2 \\ \hline \boxed{1}{,}6 \end{array}$$

primeiro algarismo ← $\boxed{1}{,}6$

Por atingir o valor, separamos a parte posterior à vírgula e reiniciamos o processo:

$$\begin{array}{r} 0{,}6 \\ \times\ 2 \\ \hline \boxed{1}{,}2 \end{array}$$

segundo algarismo ← $\boxed{1}{,}2$

Novamente, reiniciamos o processo:

$$\begin{array}{r} 0{,}2 \\ \times\ 2 \\ \hline \boxed{0}{,}4 \\ \times\ 2 \\ \hline \boxed{0}{,}8 \end{array}$$

terceiro algarismo ← $\boxed{0}{,}4$

quarto algarismo ← $\boxed{0}{,}8$

Podemos notar que o número 0,8 tornou a aparecer, logo se continuarmos o processo, teremos a mesma seqüência já vista até aqui. Este é um caso equivalente a uma dízima.

Temos, então:

$$0{,}8_{10} = (\underbrace{0{,}1100}_{\text{seqüência calculada}}\ \underbrace{1100}_{\text{repetições}}\underbrace{1100}\ldots)_2$$

logo: $4{,}8_{10} = (100{,}110011001100 1100\ldots)_2$

1.2.4.1 Exercícios Resolvidos

1 - Converta número 3,380 em binário.

Conversão da parte inteira:

$3_{10} = 11_2$

Conversão da parte fracionária:

$$
\begin{array}{ll}
 & 0,38 \\
 & \underline{\times 2} \\
1^{\circ} \leftarrow & \boxed{0},76 \\
 & \underline{\times 2} \\
2^{\circ} \leftarrow & \boxed{1},52 \\
 & 0,52 \\
 & \underline{\times 2} \\
3^{\circ} \leftarrow & \boxed{1},04 \\
 & 0,04 \\
 & \underline{\times 2} \\
4^{\circ} \leftarrow & \boxed{0},08 \\
 & \underline{\times 2} \\
5^{\circ} \leftarrow & \boxed{0},16 \\
 & \underline{\times 2} \\
6^{\circ} \leftarrow & \boxed{0},32 \\
 & \underline{\times 2} \\
7^{\circ} \leftarrow & \boxed{0},64 \\
 & \underline{\times 2} \\
8^{\circ} \leftarrow & \boxed{1},28 \\
 & 0,28 \\
 & \underline{\times 2} \\
9^{\circ} \leftarrow & \boxed{0},56
\end{array}
$$

No caso, temos:

$0{,}011000010_2 = 1 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-8} = 0{,}37890625_{10}$

Se aproximarmos o número decimal em duas casas, teremos 0,38, logo, para uma precisão de duas casas decimais é suficiente que tenhamos seguido o método até aí. Podemos escrever, então:

$0{,}38_{10} = 0{,}01100001_2 \therefore 3{,}38_{10} = 11{,}01100001_2$

Notamos que quanto mais casas considerarmos após a vírgula, teremos uma maior precisão, ou seja, devemos aplicar o método até atingir a precisão desejada.

2 - Converta o número $57{,}3_{10}$ em binário.

Conversão da parte inteira:

57 | 2
LSB ← (1) 28 | 2
(0) 14 | 2
(0) 7 | 2
(1) 3 | 2
(1) (1) → MSB

$\therefore 57_{10} = 111001_2$

Conversão da parte fracionária:

Temos, então: $0{,}3_{10} = (0{,}0100110011001...)_2$

$\therefore 57{,}3_{10} = (111001{,}0100110011001...)_2$

1.3 O Sistema Octal de Numeração

O sistema octal de numeração é um sistema de base 8 no qual existem 8 algarismos assim enumerados:

0, 1, 2, 3, 4, 5, 6 e 7

Para representarmos a quantidade oito, agimos do mesmo modo visto anteriormente para números binários e decimais, colocamos o algarismo **1** seguido do algarismo **0**, significando que temos um grupo de oito adicionado a nenhuma unidade.

Atualmente, o sistema Octal praticamente é pouco utilizado no campo da Eletrônica Digital, tratando-se apenas de um sistema numérico intermediário dos sistemas binário e hexadecimal.

Após esta introdução, podemos, utilizando o mesmo conceito, montar a seqüência de numeração do sistema para representar outras quantidades. A tabela 1.2 mostra a seqüência de numeração do sistema octal até a quantidade dezesseis.

DECIMAL	OCTAL
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	10
9	11
10	12
11	13
12	14
13	15
14	16
15	17
16	20

Tabela 1.2

1.3.1 Conversão do Sistema Octal para Sistema Decimal

Para convertermos um número octal em decimal, utilizamos o conceito básico de formação de um número, conforme já visto.

Vamos, por exemplo, converter o número 144_8 em decimal:

8^2	8^1	8^0
1	4	4

$1 \times 8^2 + 4 \times 8^1 + 4 \times 8^0 =$

$1 \times 64 + 4 \times 8 + 4 \times 1 = 64 + 32 + 4 = 100_{10}$

$\therefore\ 144_8 = 100_{10}$

1.3.1.1 Exercícios Resolvidos

1 - Converta o número 77_8 em decimal.

8^1	8^0
7	7

$7 \times 8^1 + 7 \times 8^0 = 7 \times 8 + 7 \times 1 = 56 + 7 = 63_{10}$

$\therefore\ 77_8 = 63_{10}$

2 - Converta o número 100_8 em decimal.

8^2	8^1	8^0
1	0	0

$1 \times 8^2 = 1 \times 64 = 64_{10}$ $\therefore\ 100_8 = 64_{10}$

3 - Converta o número 476_8 em decimal.

8^2	8^1	8^0
4	7	6

$4 \times 8^2 + 7 \times 8^1 + 6 \times 8^0 = 4 \times 64 + 7 \times 8 + 6 \times 1 =$

$256 + 56 + 6 = 318_{10}$ $\therefore\ 476_8 = 318_{10}$

1.3.2 Conversão do Sistema Decimal para o Sistema Octal

O processo é análogo à conversão do sistema decimal para o binário, somente que neste caso, utilizaremos a divisão por 8, pois sendo o sistema octal, sua base é igual a 8.

Para exemplificar, vamos converter o número 92_{10} para o sistema octal:

92 | 8
1º resto ← (4) 11 | 8
2º resto ← (3) (1)
último quociente ←

$\therefore 92_{10} = 134_8$

1.3.2.1 Exercícios Resolvidos

1 - Converta o número 74_{10} em octal.

$\therefore 74_{10} = 112_8$

2 - Converta o número 512_{10} em octal.

$\therefore 512_{10} = 1000_8$

3 - Converta o número 719_{10} em octal.

$\therefore 719_{10} = 1317_8$

1.3.3 Conversão de Sistema Octal para o Sistema Binário

Trata-se de uma conversão extremamente simples, podendo-se utilizar a regra prática descrita a seguir.

Vamos usar um número octal qualquer, por exemplo, o número 27_8. A regra consiste em transformar cada algarismo diretamente no correspondente em binário, respeitando-se o número padrão de bits do sistema, sendo para o octal igual a três ($2^3 = 8 \Rightarrow$ base do sistema octal). Assim sendo, temos:

$$\underbrace{010}_{2}\ \underbrace{111}_{7} \qquad \therefore\ 27_8 = 10111_2$$

Convém observar que a regra só é válida entre sistemas numéricos de base múltipla de 2^N, sendo **N** um número inteiro.

1.3.3.1 Exercícios Resolvidos

Converta os números octais em binários:

a) 34_8

$$\underbrace{011}_{3}\ \underbrace{100}_{4} \qquad \therefore\ 34_8 = 11100_2$$

b) 536_8

$$\underbrace{101}_{5}\ \underbrace{011}_{3}\ \underbrace{110}_{6} \qquad \therefore\ 536_8 = 101011110_2$$

c) 44675_8

$$\underbrace{100}_{4}\ \underbrace{100}_{4}\ \underbrace{110}_{6}\ \underbrace{111}_{7}\ \underbrace{101}_{5} \qquad \therefore\ 44675_8 = 100100110111101_2$$

1.3.4 Conversão do Sistema Binário para o Sistema Octal

Para efetuar esta conversão, vamos aplicar o processo inverso ao utilizado na conversão de octal para binário. Como exemplo, vamos utilizar o número 110010_2.

Para transformar este número em octal, vamos primeiramente separá-lo em grupos de 3 bits a partir da direita:

110 010

Efetuando, agora, a conversão de cada grupo de bits diretamente para o sistema octal, temos:

$$\underbrace{110}_{6}\ \underbrace{010}_{2}$$

O número convertido será composto pela união dos algarismos obtidos.

$\therefore\ 110010_2 = 62_8$

No caso do último grupo se formar incompleto, adicionamos zeros à esquerda, até completá-lo com 3 bits. Para exemplificar, vamos converter o número 1010_2 em octal:

1 010

Acrescentamos zeros à esquerda, até completar o grupo de 3 bits. A partir daí, utilizamos o processo já visto:

$\underbrace{001}_{1}\ \underbrace{010}_{2} \qquad \therefore 1010_2 = 12_8$

1.3.4.1 Exercícios Resolvidos

Converta os números binários em octais:

a) 10111_2

Vamos separar o número em grupos de 3 bits a partir da direita e efetuar a conversão:

$\underbrace{010}_{2}\ \underbrace{111}_{7} \qquad \therefore 10111_2 = 27_8$

b) 11010101_2

$\underbrace{011}_{3}\ \underbrace{010}_{2}\ \underbrace{101}_{5} \qquad \therefore 11010101_2 = 325_8$

c) 1000110011_2

$\underbrace{001}_{1}\ \underbrace{000}_{0}\ \underbrace{110}_{6}\ \underbrace{011}_{3} \qquad \therefore 1000110011_2 = 1063_8$

1.4 O Sistema Hexadecimal de Numeração

O sistema hexadecimal possui 16 algarismos, sendo sua base igual a 16. Os algarismos são assim enumerados:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, e F

Notamos que a letra A representa o algarismo **A,** que por sua vez representa a quantidade dez. A letra B representa o algarismo **B** que representa a quantidade onze, e assim sucede até a letra **F** que representa a quantidade quinze.

Para representarmos a quantidade dezesseis, utilizamos o conceito básico da formação de um número, ou seja, colocamos o algarismo **1** seguido do algarismo **0**, representando um grupo de dezesseis adicionado a nenhuma unidade.

Após esta introdução, podemos formar a seqüência de numeração hexadecimal. A tabela 1.3 mostra a seqüência de numeração do sistema hexadecimal até a quantidade vinte.

DECIMAL	HEXADECIMAL
0	0
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	A
11	B
12	C
13	D
14	E
15	F
16	10
17	11
18	12
19	13
20	14

Tabela 1.3

Este sistema é muito utilizado na área dos microprocessadores e também no mapeamento de memórias em sistemas digitais, tratando-se de um sistema numérico muito importante, sendo aplicado em projetos de **software** e **hardware**.

1.4.1 Conversão do Sistema Hexadecimal para o Sistema Decimal

A regra de conversão é análoga à de outros sistemas, somente que neste caso, a base é 16. Como exemplo, vamos utilizar o número $3F_{16}$ e convertê-lo em decimal:

16^1	16^0
3	F

$3 \times 16^1 + F \times 16^0 =$

sendo $F_{16} = 15_{10}$, substituindo temos:

$3 \times 16^1 + 15 \times 16^0 = 3 \times 16 + 15 \times 1 = 63_{10}$

$\therefore 3F_{16} = 63_{10}$

1.4.1.1 Exercícios Resolvidos

Converta os números em hexadecimal para decimal:

a) $1C3_{16}$

16^2	16^1	16^0
1	C	3

$1 \times 16^2 + C \times 16^1 + 3 \times 16^0 =$

Substituindo C_{16} por 12_{10}, temos:

$1 \times 16^2 + 12 \times 16^1 + 3 \times 16^0 = 1 \times 256 + 12 \times 16 + 3 \times 1 = 451_{10}$

$\therefore 1C3_{16} = 451_{10}$

b) 238_{16}

16^2	16^1	16^0
2	3	8

$2 \times 16^2 + 3 \times 16^1 + 8 \times 16^0 =$

$2 \times 256 + 3 \times 16 + 8 \times 1 = 568_{10}$

$\therefore 238_{16} = 568_{10}$

c) $1FC9_{16}$

16^3	16^2	16^1	16^0
1	F	C	9

$1 \times 16^3 + F \times 16^2 + C \times 16^1 + 9 \times 16^0 =$

$1 \times 16^3 + 15 \times 16^2 + 12 \times 16^1 + 9 \times 16^0 =$

$1 \times 4096 + 15 \times 256 + 12 \times 16 + 9 \times 1 = 8137_{10}$

$\therefore 1FC9_{16} = 8137_{10}$

1.4.2 Conversão do Sistema Decimal para o Sistema Hexadecimal

Da mesma forma que nos casos anteriores, esta conversão se faz através de divisões sucessivas pela base do sistema a ser convertido. Para exemplificar vamos transformar o número 1000_{10} em hexadecimal:

Sendo $14_{10} = E_{16}$, temos: $3E8_{16}$

$\therefore 1000_{10} = 3E8_{16}$

1.4.2.1 Exercícios Resolvidos

1 - Converta o número 134_{10} para o sistema hexadecimal.

134 | 16
6   8

$\therefore 134_{10} = 86_{16}$

2 - Idem ao anterior, para o número 384_{10}.

384 | 16
0   24 | 16
    8   1

$\therefore 384_{10} = 180_{16}$

3 - Converta o número 3882_{10} em hexadecimal.

$$\begin{array}{l} 3882 \,\lfloor 16 \\ (10)\; 242 \,\lfloor 16 \\ \quad\quad (2)\;(15) \end{array} \quad \therefore 3882_{10} = F2A_{16}$$

1.4.3 Conversão do Sistema Hexadecimal para o Sistema Binário

É análoga à conversão do sistema octal para o sistema binário, somente que, neste caso, necessita-se de 4 bits para representar cada algarismo hexadecimal.

Como exemplo, vamos converter o número $C13_{16}$ para o sistema binário:

$$\underbrace{C}_{1100} \Rightarrow (C_{16} = 12_{10}) \quad \underbrace{1}_{0001} \quad \underbrace{3}_{0011}$$

$\therefore C13_{16} = 110000010011_2$

1.4.3.1 Exercícios Resolvidos

1 - Converta para o sistema binário:

a) $1ED_{16}$

$$\underbrace{1}_{0001} \quad \underbrace{E}_{1110} \Rightarrow (E_{16} = 14_{10}) \qquad \underbrace{D}_{1101} \Rightarrow (D_{16} = 13_{10})$$

$\therefore 1ED_{16} = 111101101_2$

b) $6CF9_{16}$

$$\underbrace{6}_{0110} \quad \underbrace{C}_{1100} \quad \underbrace{F}_{1111} \quad \underbrace{9}_{1001}$$

$6CF9_{16} = 110110011111001_2$

2 - Converta o número $3A7_{16}$ para o sistema octal.

A solução é simples, bastando converter o número para o sistema binário e logo após, da forma já vista, agrupar o resultado de 3 em 3 bits, obtendo então o resultado em octal. Assim sendo, temos:

$$\overbrace{0011}^{3}\ \overbrace{1010}^{A}\ \overbrace{0111}^{7}$$

temos, então: $3A7_{16} = 1110100111_2$

Agora, transformando diretamente para octal, temos:

$$\underbrace{001}_{1}\underbrace{110}_{6}\underbrace{100}_{4}\underbrace{111}_{7}\ \therefore\ 3A7_{16} = 1647_8$$

1.4.4 Conversão do Sistema Binário para o Sistema Hexadecimal

É análoga à conversão do sistema binário para o octal, somente que neste caso, agrupamos de 4 em 4 bits da direita para a esquerda. A título de exemplo, vamos transformar o número 10011000_2 em hexadecimal:

$$\underbrace{1001}_{9}\underbrace{1000}_{8}\ \therefore\ 10011000_2 = 98_{16}$$

1.4.4.1 Exercícios Resolvidos

Converta para o sistema hexadecimal os números binários:

a) 1100011_2

$$\underbrace{0110}_{6}\ \underbrace{0011}_{3}\ \therefore\ 1100011_2 = 63_{16}$$

b) 11000111100011100_2

$$\underbrace{0001}_{1}\ \underbrace{1000}_{8}\ \underbrace{1111}_{F}\ \underbrace{0001}_{1}\ \underbrace{1100}_{C}$$

$\therefore\ 11000111100011100_2 = 18F1C_{16}$

1.5 Operações Aritméticas no Sistema Binário

Nas áreas da Eletrônica Digital e dos Microprocessadores, o estudo das operações aritméticas no sistema binário é muito importante, pois estas serão utilizadas em circuitos aritméticos, tópico este, que será visto em capítulo mais adiante.

1.5.1 Adição no Sistema Binário

Para efetuarmos a adição no sistema binário, devemos agir como numa adição convencional no sistema decimal, lembrando que, no sistema binário, temos apenas 2 algarismos. Temos, então:

$$\begin{array}{r}0\\+0\\\hline 0\end{array}\quad\begin{array}{r}0\\+1\\\hline 1\end{array}\quad\begin{array}{r}1\\+0\\\hline 1\end{array}\quad\begin{array}{r}1\\+1\\\hline 10\end{array}$$

Convém observar que no sistema decimal 1 + 1 = 2 e no sistema binário representamos o número 2_{10} por 10_2. Pela operação realizada, notamos a regra de transporte para a próxima coluna: 1 + 1 = 0 e transporta 1 "vai um".

A operação de transporte também é denominada **carry**, termo derivado do inglês.

Para exemplificar, vamos somar os números binários 11_2 e 10_2.

Vamos efetuar a adição coluna a coluna, considerando o transporte proveniente da coluna anterior:

$$\begin{array}{r}1\\+11\\10\\\hline 101\end{array}$$

1+1=0 e transporta 1

$\therefore\ 11_2 + 10_2 = 101_2$

Verificação: $(3_{10} + 2_{10} = 5_{10})$

Como outro exemplo, vamos efetuar $110_2 + 111_2$:

$$\begin{array}{r}11\\+110\\111\\\hline 1101\end{array}$$

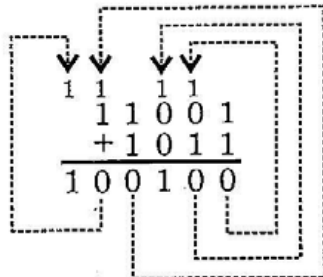
1+1+1= 11

$\therefore\ 110_2 + 111_2 = 1101_2$

Verificação: $(6_{10} + 7_{10} = 13_{10})$

1.5.1.1 Exercícios Resolvidos

1 - Efetue as operações no Sistema binário

a) $11001_2 + 1011_2$:

$\therefore 11001_2 + 1011_2 = 100100_2$

b) $101101_2 + 11100011_2$:

```
111  1111      ──→    transportes
    101101
+11100011
100010000
```

$\therefore 101101_2 + 11100011_2 = 100010000_2$

c) $11111_2 + 111111_2$:

```
111111        ──→           transportes
  11111
+111111
1011110
```

$\therefore 11111_2 + 111111_2 = 1011110_2$

d) $100111_2 + 1110_2 + 1011_2$:

Neste exemplo, observamos que o transporte acumulado para a coluna seguinte é 10, pois 1 + 1 + 1 + 1 = 100.

$\therefore 100111_2 + 1110_2 + 1011_2 = 1000000_2$

1.5.2 Subtração no Sistema Binário

O método de resolução é análogo a uma subtração no sistema decimal. Temos, então:

$$\begin{array}{cccc} 0 & 0 & 1 & 1 \\ -0 & -1 & -0 & -1 \\ \hline 0 & 1 & 1 & 0 \end{array}$$

Observamos que para o caso 0-1, o resultado será igual a 1, porém haverá um transporte para a coluna seguinte que deve ser acumulado no subtraendo e, obviamente, subtraído do minuendo.

Para exemplificar, vamos efetuar a operação $111_2 - 100_2$:

$$\begin{array}{ccc} 1 & 1 & 1 \\ -1 & 0 & 0 \\ \hline 0 & 1 & 1 \end{array}$$

$\therefore 111_2 - 100_2 = 11_2$ $(7_{10} - 4_{10} = 3_{10})$

Agora, para melhor elucidar o caso 0-1, vamos resolver a operação $1000_2 - 111_2$ passo a passo. Assim sendo, temos:

$$\begin{array}{r} 1\,0\,0\,0 \\ -\;1\,1\,1 \\ \hline 1 \end{array}$$

0-1=1 e transporta 1 para a coluna seguinte

```
 1 0 0 0       --- transporte anterior
-   1 1
  1 1 1     0-1-1=0 e transporta 1 para
 -------
    0 1     a coluna seguinte
```

```
 1 0 0 0
-1 1
  1 1 1     0-1-1=0 e transporta 1 para
 -------
  0 0 1     a coluna seguinte
```

```
 1 0 0 0
-1
  1 1 1          transporte
 -------         anterior
 0 0 0 1
 ---------> 1-1=0
```

$\therefore 1000_2 - 111_2 = 0001_2$

1.5.2.1 Exercícios Resolvidos

1 - Efetue as operações no sistema binário:

a) $1010_2 - 1000_2$

1	0	1	0
-1	0	0	0
0	0	1	0

$\therefore 1010_2 - 1000_2 = 10_2$

b) $10010_2 - 10001_2$

```
 1 0 0 1 0
-      1
 1 0 0 0 1   0-1=1 e transporta 1
 ---------
 0 0 0 0 1
```

$\therefore 10010_2 - 10001_2 = 1_2$

c) $11000_2 - 111_2$

```
 1 1 0 0 0
   1 1 1
-    1 1 1
 ---------
 1 0 0 0 1
```

$\therefore 11000_2 - 111_2 = 10001_2$

1.5.3 Multiplicação no Sistema Binário

Procede-se como em uma multiplicação no sistema decimal. Assim sendo, temos:

0 x 0 = 0

0 x 1 = 0

1 x 0 = 0

1 x 1 = 1

Para exemplificar, vamos efetuar a operação $11010_2 \times 10_2$:

```
 11010
   x10
------
 00000
11010+
------
110100
```

$\therefore 11010_2 \times 10_2 = 110100_2$

1.5.3.1 Exercício Resolvido

1 - Efetue as multiplicações no sistema binário:

a) $1100_2 \times 011_2$:

```
  1100
   x11
------
  1100
 1100+
------
100100
```

$\therefore 1100_2 \times 011_2 = 100100$

b) $11010_2 \times 101_2$:

```
   11010
    x101
--------
   11010
  00000+
 11010+
--------
10000010
```

$\therefore 11010_2 \times 101_2 = 10000010$

c) $100101_2 \times 1001_2$:

```
     100101
      x1001
  ---------
     100101
    000000
   000000
  100101   +
  ---------
  101001101
```

$\therefore\ 100101_2 \times 1001_2 = 101001101_2$

Nota:	A divisão de números binários é a mais complexa das operações aritméticas binárias, pois abrange operações de multiplicação e subtração. Não vamos abordá-la neste capítulo, pois não a utilizaremos no estudo dos circuitos digitais.

1.5.4 Notação dos Números Binários Positivos e Negativos

A representação de números binários positivos e negativos pode ser feita utilizando-se os sinais "+" ou "-" respectivamente. Na prática, porém, em hardware dos sistemas digitais que processam operações aritméticas, microcomputadores por exemplo, estes sinais não podem ser utilizados, pois tudo deve ser codificado em 0 ou 1. Uma forma de representar em alguns casos utilizada, é a de acrescentar ao número um **bit de Sinal** colocado à esquerda, na posição de algarismo mais significativo. Se o número for positivo, o bit de sinal será **0**, se o número for negativo este será 1. Este processo de representação é denominado **Sinal-módulo.**

Para exemplificar o exposto, vamos representar os números decimais $+35_{10}$ e -73_{10} em binário utilizando a notação sinal-módulo:

$35_{10} = 100011_2$

$\therefore\ +100011_2 = \mathbf{0}100011_2$

└→ bit de sinal (0 → indica número positivo)

$73_{10} = 1001001_2$

$\therefore\ -1001001_2 = \mathbf{1}1001001_2$

└→ bit de sinal (1 → indica número negativo)

Uma outra forma para representar número binários negativos bastante utilizada nos sistemas já citados é a notação **do complemento de 2,** mas para obtê-la, devemos primeiramente converter o número na notação do **complemento de 1**, conforme se segue.

A obtenção do complemento de 1 de um número binário se dá pela troca de cada bit do número pelo seu inverso ou complemento. Para demonstrar esse procedimento, vamos obter o complemento de 1 do número 10011011_2. Assim sendo, temos:

Número binário: 1 0 0 1 1 0 1 1

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

Complemento de 1: 0 1 1 0 0 1 0 0

∴ O complemento de 1 de 10011011_2 é 01100100_2.

A notação do complemento de 2, como já dissemos, é utilizada para representar números binários negativos. Sua obtenção se dá somando-se 1 ao complemento de 1 do número binário inicial. Para exemplificar, vamos representar o número -11001101_2 na notação do complemento de 2:

Número binário:	11001101
Complemento de 1:	00110010
	+ 1
Complemento de 2:	00110011

∴ A representação na notação do complemento de 2 do número -11001101_2 é 00110011_2.

Convém observar que estas representações, por serem utilizadas no hardware de sistemas, possuem sempre um número predefinido de bits, não devendo ser desconsiderado nenhum deles na resposta. Nos exemplos já vistos, utilizamos números com 8 bits.

Utilizando estas definições, vamos mostrar, a título de exemplo, uma tabela representativa da seqüência de números binários positivos e negativos. A tabela 1.4 mostra a representação dos números -9_{10} a $+9_{10}$ no sistema binário em 4 bits utilizando a notação do complemento de 2.

DECIMAL	-9	-8	-7	-6	-5	-4	-3	-2	-1
BINÁRIO	-1001	-1000	-0111	-0110	-0101	-0100	-0011	-0010	-0001
COMPL. DE 2	0111	1000	1001	1010	1011	1100	1101	1110	1111

Tabela 1.4 (parte)

DECIMAL	0	+1	+2	+3	+4	+5	+6	+7	+8	+9
BINÁRIO	0000	+0001	+0010	+0011	+0100	+0101	+0110	+0111	+1000	+1001
COMPL. DE 2	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Tabela 1.4

Pela tabela, notamos que os números positivos na notação do complemento de 2 recebem representação normal, idêntica à do sistema binário. Nos sistemas digitais, para se efetuar uma diferenciação, utiliza-se da mesma forma, um bit de sinal a mais que colocado à esquerda do número, indica se este é positivo (bit de sinal = 0) ou se este é negativo (bit de sinal = 1), estando na notação do complemento de 2.

Um outro ponto, de grande importância, a ser abordado é a conversão inversa, ou seja, a passagem do número na notação do complemento de 2 para a notação binária normal. O processo é simples, bastando determinarmos novamente o complemento de 2 do resultado.

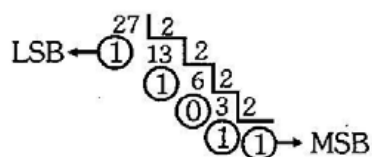
Para esclarecermos melhor, vamos aplicar este processo ao número 1011_2 (na notação do complemento de 2) que de acordo com a tabela 1.4 significa -0101_2 ou -5_{10}. Assim sendo, temos:

	1011
Complemento de 1:	0100
	+ 1
Complemento de 2:	0101

∴ Extraindo novamente o complemento de 2 do número 1011_2 obtemos -0101_2 (-5_{10}), que é o mesmo número, porém, na notação normal.

1.5.4.1 Exercícios Resolvidos

1 - Represente os seguintes números utilizando a notação sinal-módulo.

a) -27_{10}

$27_{10} = 11011_2$

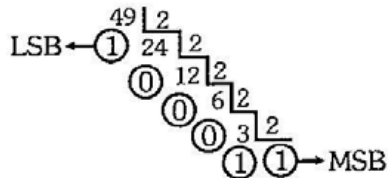
Sendo o número negativo, para representar na notação sinal-módulo acrescentamos 1 como o bit de sinal à esquerda do número:

111011

$\therefore$ $-27_{10} = \mathbf{1}11011_2$

b) $+49_{10}$

Da mesma forma, temos:

49 | 2
LSB ← (1) 24 | 2
(0) 12 | 2
(0) 6 | 2
(0) 3 | 2
(1) (1) → MSB

$49_{10} = 110001_2 \Rightarrow \mathbf{0}110001_2$ (em sinal-módulo)

$\therefore$ $+49_{10} = \mathbf{0}110001_2$

2 - Determine o complemento de 1 do número 101010_2.

Invertendo o número bit a bit, temos:

101010 ⇒ 010101

$\therefore$ O complemento de 1 de 101010_2 é 010101_2.

3 - Determine o complemento de 2 do número -10010110_2.

Seguindo o processo, temos:

	10010110
Complemento de 1:	01101001
	+1
Complemento de 2:	01101010

$\therefore$ O número -10010110_2 em complemento de 2 é 01101010_2.

4 - Qual o equivalente positivo do número 0110_2, aqui representado em complemento de 2? Para solucionar basta determinarmos novamente o complemento de 2. Assim sendo, temos:

	0110
Complemento do 1:	1001
	+ 1
Complemento do 2:	1010

∴ O equivalente positivo do número 0110_2 (em complemento de 2) é 1010_2.

1.5.5 Utilização do Complemento de 2 em Operações Aritméticas

Podemos utilizar a notação do complemento de 2 para efetuar operações diversas que envolvam soma ou subtração. De maneira geral, podemos considerá-las como operações de soma envolvendo números positivos e negativos, ou ainda entre números quaisquer, obtendo uma resposta apropriada conforme a situação.

Para solucionar qualquer operação destas, basta determinar o complemento de 2 do número negativo envolvido, com o mesmo número de bits do outro membro da operação e realizar a soma, desconsiderando, se houver, o estouro do número de bits no resultado.

A título de exemplo, vamos efetuar a operação $11010111_2 - 100101_2$.

Notamos que esta operação equivale à soma de um número binário positivo com outro negativo: $N_1 + (-N_2)$. Como vimos, a solução se dá determinando o complemento de 2 do segundo (negativo) com mesmo número de bits do primeiro, efetuando a soma e eliminando o bit em excesso. Procedendo assim, temos:

$11010111_2 - 100101$:

Complemento de 1 de 00100101: 11011010

Complemento de 2:	11011010
	+ 1
	11011011
Operação:	11010111
	+11011011
	X10110010

└→ estouro do número de bits desconsiderado

∴ $11010111_2 - 100101_2 = 10110010_2$

A vantagem deste processo é que nos sistemas digitais pode-se utilizar um mesmo circuito somador para efetuar-se operações que envolvam números negativos ou ainda subtrações, simplificando a quantidade de componentes no sistema. Utilizaremos também estes conceitos no capítulo relativo a circuitos aritméticos.

1.5.5.1 Exercícios Resolvidos

1 - Efetue as subtrações, utilizando o complemento de 2:

a) $10101011_2 - 1000100_2$

Seguindo o mesmo processo, temos:

Complemento de 1 de 01000100: 10111011

Complemento de 2:

```
  10111011
+        1
----------
  10111100
```

Operação:

```
  10101011
 +10111100
----------
 X01100111
 └→ estouro desconsiderado
```

$\therefore 10101011_2 - 1000100_2 = 1100111_2$

b) $10011_2 - 100101_2$

Trata-se de um número menor subtraindo um outro maior. Agindo da mesma forma, temos:

Complemento de 1 de 100101: 011010

Complemento de 2: 011011

Operação:

```
  010011
 +011011
--------
  101110
```

Pelo fato de o minuendo (10011_2) ser menor que o subtraendo (100101_2) a resposta é negativa, estando na notação do complemento de 2. Para obtê-la na notação binária normal, basta determinar novamente o complemento de 2 e acrescentar o sinal negativo:

$101110 \Rightarrow 010001 \Rightarrow 010001 + 1 = 010010$

$\therefore 10011_2 - 100101_2 = -10010_2$ (ou 101110 em complemento de 2)

2 - Efetue em binário, utilizando a aritmética do complemento de 2, a operação CA_{16} - $7D_{16}$.

Para solucionar, convertemos primeiramente os números hexadecimais em binários:

$CA_{16} = 11001010_2$ e $7D_{16} = 01111101_2$

Logo após, aplicamos o mesmo processo já visto:

Complemento de 2 de 01111101:

$10000010 \Rightarrow 10000010 + 1 \Rightarrow 10000011$

Operação:

$$\begin{array}{r} 11001010 \\ +10000011 \\ \hline \not{1}01001101 \end{array}$$

Após obtido o resultado, o transformamos em hexadecimal:

$01001101_2 = 4D_{16}$.

$\therefore CA_{16} - 7D_{16} = 4D_{16}$

1.6 Exercícios Propostos

1.6.1 - Converta para o sistema decimal:

a) 100110_2
b) 011110_2
c) 111011_2
d) 1010000_2
e) 11000101_2
f) 11010110_2
g) 011001100110101_2

1.6.2 - Converta para o sistema binário:

a) 78_{10}
b) 102_{10}
c) 215_{10}
d) 404_{10}
e) 808_{10}
f) 5429_{10}
g) 16383_{10}

1.6.3 - Quantos bits necessitaríamos para representar cada um dos números decimais abaixo?

a) 512_{10}
b) 12_{10}
c) 2_{10}
d) 17_{10}
e) 33_{10}
f) 43_{10}
g) 7_{10}

1.6.4 - Transforme para decimal os seguintes números binários:

a) $11,11_2$
b) $1000,0001_2$
c) $1010,1010_2$
d) $1100,1101_2$
e) $10011,10011_2$
f) $11000,001101_2$
g) $100001,011001_2$

1.6.5 - Transforme os seguintes números decimais em binários:

a) $0{,}125_{10}$
b) $0{,}0625_{10}$
c) $0{,}7_{10}$
d) $0{,}92_{10}$
e) $7{,}9_{10}$
f) $47{,}47_{10}$
g) $53{,}3876_{10}$

1.6.6 - Transforme os números octais para o sistema decimal:

a) 14_8
b) 67_8
c) 153_8
d) 1544_8
e) 2063_8

1.6.7 - Por que o número 15874 não pode ser octal?

1.6.8 - Converta para o sistema octal:

a) 107_{10}
b) 185_{10}
c) 2048_{10}
d) 4097_{10}
e) 5666_{10}

1.6.9 - Converta os seguintes números octais em binários:

a) 477_8 **d)** 6740_8

b) 1523_8 **e)** 10021_8

c) 4764_8

1.6.10 - Converta os seguintes números binários em octais:

a) 1011_2 **d)** 1000000001_2

b) 10011100_2 **e)** 1101000101_2

c) 110101110_2

1.6.11 - Converta para o sistema decimal os seguintes números hexadecimais:

a) 479_{16} **d)** $FOCA_{16}$

b) $4AB_{16}$ **e)** $2D3F_{16}$

c) BDE_{16}

1.6.12 - Converta os seguintes números decimais em hexadecimais:

a) 486_{10} **d)** 5555_{10}

b) 2000_{10} **e)** 35479_{10}

c) 4096_{10}

1.6.13 - Converta para o sistema binário:

a) 84_{16} **d)** $47FD_{16}$

b) $7F_{16}$ **e)** $F1CD_{16}$

c) $3B8C_{16}$

1.6.14 - Converta os números $1D2_{16}$ e $8CF_{16}$ para o sistema octal.

1.6.15 - Converta para o sistema hexadecimal os seguintes números binários:

a) 10011_2 **d)** 11111011110010_2

b) 1110011100_2 **e)** 1000000000100010_2

c) 100110010011_2

1.6.16 - Converta os números 7100_8 e 5463_8 para hexadecimal.

1.6.17 - Efetue as operações:

a) $1000_2 + 1001_2$

b) $10001_2 + 11110_2$

c) $101_2 + 100101_2$

d) $1110_2 + 1001011_2 + 11101_2$

e) $110101_2 + 1011001_2 + 1111110_2$

1.6.18 - Resolva as subtrações, no sistema binário:

a) $1100_2 - 1010_2$

b) $10101_2 - 1110_2$

c) $11110_2 - 1111_2$

d) $1011001_2 - 11011_2$

e) $100000_2 - 11100_2$

1.6.19 - Multiplique:

a) $10101_2 \times 11_2$

b) $11001_2 \times 101_2$

c) $110110_2 \times 111_2$

d) $11110_2 \times 110_2$

e) $100110_2 \times 1010_2$

1.6.20 - Represente os números $+97_{10}$ e -121_{10}, utilizando a notação sinal -módulo.

1.6.21 - Estando o número 10110010 em sinal-módulo, o que ele representa no sistema decimal?

1.6.22 - Determine o complemento de 1 de cada número binário:

a) 01110100_2

b) 11000010_2

1.6.23 - Represente os seguintes números na notação do complemento de 2:

a) -1011_2

b) -100001_2

c) -10111101_2

d) -11010100_2

e) -01010011_2

1.6.24 - Qual o equivalente em decimal do número 10110111_2, aqui representado em complemento de 2?

1.6.25 - Efetue as operações utilizando o complemento de 2:

a) $101101_2 - 100111_2$

b) $10000110_2 - 110011_2$

c) $111100_2 - 11101011_2$

d) $-10010011_2 + 11011010_2$

e) $-10011101 - 1000101_2$

1.6.26 - Efetue em binário as operações, utilizando a aritmética do complemento de 2:

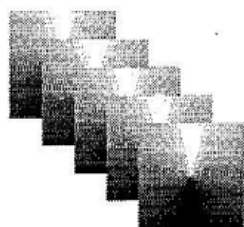
a) $75_8 - 30_8$

b) $44_{16} - 3E_{16}$

c) $A9_{16} - EO_{16}$

d) $-BC_{16} + FC_{16}$

e) $-22_{16} - 1D_{16}$

CAPÍTULO 2

Funções e Portas Lógicas

2.1 Introdução

Em 1854, o matemático inglês **George Boole** (1815 - 1864), através da obra intitulada **An Investigation of the Laws of Thought**, apresentou um sistema matemático de análise lógica conhecido como **álgebra de Boole**.

No início da "era da eletrônica", todos os problemas eram resolvidos por sistemas analógicos, também conhecidos por sistemas lineares.

Apenas em 1938, o engenheiro americano **Claude Elwood Shannon** utilizou as teorias da álgebra de Boole para a solução de problemas de circuitos de telefonia com relés, tendo publicado um trabalho denominado **Symbolic Analysis of Relay and Switching**, praticamente introduzindo na área tecnológica o campo da eletrônica digital.

Esse ramo da eletrônica emprega em seus sistemas um pequeno grupo de circuitos básicos padronizados conhecidos como portas lógicas.

Através da utilização conveniente destas portas, podemos "implementar" todas as expressões geradas pela álgebra de Boole, que constituem a base dos projetos dos sistemas já referidos.

Neste capítulo, trataremos apenas dos blocos básicos, deixando para os capítulos posteriores, estudo de outros blocos e sistemas derivados.

2.2 Funções Lógicas E, OU, NÃO, NE e NOU

Faremos, a seguir, o estudo das principais funções lógicas que na realidade derivam dos postulados da álgebra de Boole, sendo as variáveis e expressões envolvidas denominadas de booleanas.

Nas funções lógicas, temos apenas dois estados distintos:

➪ o estado **0** (zero) e

➪ o estado **1** (um).

O estado 0 representará, por exemplo: portão fechado, aparelho desligado, ausência de tensão, chave aberta, não, etc. O estado 1 representará, então: portão aberto, aparelho ligado, presença de tensão, chave fechada, sim, etc.

Note, então, que se representarmos por 0 uma situação, representamos por 1 a situação contrária. Deve-se salientar aqui, que cada variável booleana da função lógica pode assumir somente 2 situações distintas 0 ou 1.

2.2.1 Função E ou AND

A função **E** é aquela que executa a multiplicação de 2 ou mais variáveis booleanas. É também conhecida como função **AND**, nome derivado do inglês. Sua representação algébrica para 2 variáveis é S = A . B, onde se lê S = A **e** B.

Para melhor compreensão, vamos utilizar e analisar o circuito representativo da função E visto na Figura 2.1.

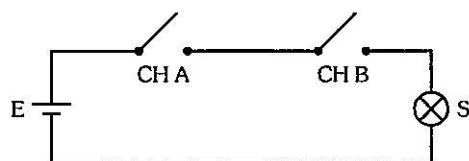



Figura 2.1

Convenções:	chave aberta = 0	chave fechada = 1
	lâmpada apagada = 0	lâmpada acesa = 1

Situações possíveis:

1º) Se tivermos a chave A aberta (0) e a chave B aberta (0), neste circuito não circulará corrente, logo, a lâmpada permanecerá apagada (0): A = 0, B = 0 ⇒ S = A.B = 0.

2º) Se tivermos a chave A aberta (0) e a chave B fechada (1), logo a lâmpada permanecerá apagada (0): $A = 0, B = 1 \Rightarrow S = 0$.

3º) Se tivermos a chave A fechada (1) e a chave B aberta (0), a lâmpada permanecerá apagada: $A = 1, B = 0 \Rightarrow S = 0$.

4º) Se tivermos, agora, a chave A fechada (1) e a chave B fechada (1), a lâmpada irá acender, pois circulará corrente: $A = 1, B = 1 \Rightarrow S = 1$.

Analisando as situações, concluímos que só teremos a lâmpada acesa quando as chaves A e B estiverem fechadas.

2.2.1.1 Tabela da Verdade de uma Função E ou AND

Chamamos **Tabela da Verdade** um mapa onde colocamos todas as possíveis situações com seus respectivos resultados. Na tabela, iremos encontrar o modo como a função se comporta. A seguir, iremos apresentar a tabela da verdade de uma função E ou AND para 2 variáveis de entrada:

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 2.1

2.2.1.2 Porta E ou AND

A porta E é um circuito que executa a função E, sendo representada na prática, através do símbolo visto na figura 2.2.

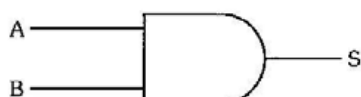



Figura 2.2

Como já dissemos, a porta E executa a tabela da verdade da função E, ou seja, teremos a saída no estado 1 se, e somente se, as 2 entradas forem iguais a 1, e teremos a saída igual a 0 nos demais casos.

Até agora, descrevemos a função E para 2 variáveis de entrada. Podemos estender esse conceito para qualquer número de entradas. Para exemplificar, mostraremos uma porta E de 3 variáveis de entrada, sua tabela da verdade, e ainda, sua expressão booleana:

S=A.B.C

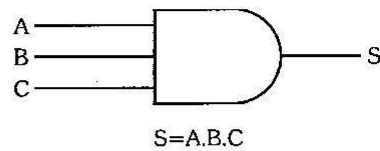



Figura 2.3

A	B	C	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Tabela 2.2

Notamos que a tabela da verdade mostra as 8 possíveis combinações das variáveis de entrada e seus respectivos resultados na saída.

O número de situações possíveis é igual a 2^N, onde N é o número de variáveis de entrada. No exemplo: $N = 3 \therefore 2^3 = 8$.

2.2.2 Função OU ou OR

A função OU é aquela que assume valor 1 quando uma ou mais variáveis da entrada forem iguais a 1 e assume valor 0 se, e somente se, todas as variáveis de entrada forem iguais a 0. Sua representação algébrica para 2 variáveis de entrada é S = A + B, onde se lê S = A **ou** B.

O termo OR, também utilizado, é derivado do inglês.

Para entendermos melhor a função OU, vamos representá-la através do circuito da figura 2.4 e analisar as situações possíveis.

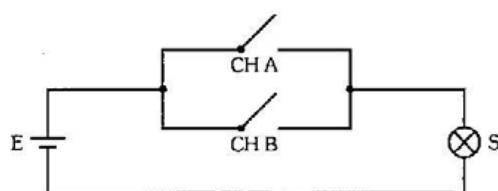



Figura 2.4

Usaremos as mesmas convenções do circuito representativo da função E, visto anteriormente. Situações possíveis:

1º) Se tivermos a chave A aberta (0) e a chave B aberta (0), no circuito não circulará corrente, logo, a lâmpada permanecerá apagada (0): A = 0, B = 0 ⇒ S = A + B = 0.

2º) Se tivermos a chave A aberta (0) e a chave B fechada (1), circulará corrente pela chave B e a lâmpada acenderá (1): A = 0, B = 1 ⇒ S = 1.

3º) Se tivermos a chave A fechada (1) e a chave B aberta (0), circulará corrente pela chave A e a lâmpada acenderá (1): A = 1, B = 0 ⇒ S = 1.

4º) Se tivermos a chave A fechada (1) e a chave B fechada (1), circulará corrente pelas duas chaves e a lâmpada acenderá (1): A = 1, B = 1 ⇒ S = 1.

Para o caso A = 1 e B = 1, a soma A + B = 1, a princípio estranha, é verdadeira, pois, como veremos mais à frente, trata-se de uma soma booleana derivada do postulado da adição da álgebra de Boole.

Notamos pelas situações que teremos a lâmpada ligada quando chA **ou** chB **ou** ambas as chaves estiverem ligadas.

2.2.2.1 Tabela da Verdade da Função OU ou OR

Nesta tabela da verdade, teremos todas as situações possíveis com os respectivos valores que a função OU assume. A tabela 2.3 apresenta a tabela da verdade da função OU ou OR para 2 variáveis de entrada.

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

Tabela 2.3

2.2.2.2 Porta OU ou OR

É a porta que executa a função OU . Representaremos a porta OU através do símbolo visto na figura 2.5.

Figura 2.5

A porta OU executa a tabela da verdade de função OU, ou seja, teremos a saída igual a 1 quando uma ou mais variáveis de entrada forem iguais a 1 e 0 se, e somente se, todas as variáveis de entrada forem iguais a 0.

Podemos estender o conceito para mais de 2 variáveis de entrada. Como exemplo, vamos mostrar uma porta OU, sua tabela da verdade e sua expressão booleana com 4 variáveis de entrada:

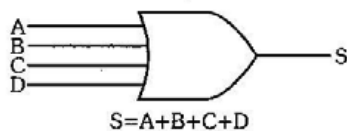



Figura 2.6

A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Tabela 2.4

Notamos, pela tabela, que as 4 variáveis de entrada possibilitam 16 combinações possíveis ($2^4 = 16$).

2.2.3 Função NÃO ou NOT

A função NÃO é aquela que inverte ou complementa o estado da variável, ou seja, se a variável estiver em 0, a saída vai para 1, e se estiver em 1, a saída vai para 0. É representada algebricamente da seguinte forma: $S = \overline{A}$ ou $S = A'$, onde se lê **A barra** ou **NÃO A**.

Esta barra ou apóstrofo sobre a letra que representa a variável, significa que esta sofre uma inversão. Também, podemos dizer que $\overline{A}$ significa a negação de A.

Para entendermos melhor a função NÃO vamos representá-la pelo circuito da figura 2.7. Analisaremos utilizando as mesmas convenções dos casos anteriores.

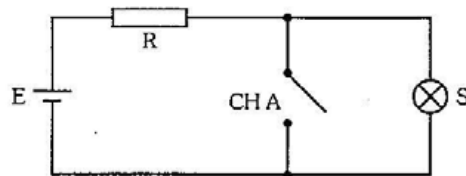



Figura 2.7

Situações possíveis:

1º) Quando a chave A estiver aberta (0), passará corrente pela lâmpada e esta acenderá (1): $A = 0 \Rightarrow S = \overline{A} = 1$.

2º) Quando a chave A estiver fechada (1), curto-circuitaremos a lâmpada e esta se apagará (0): $A = 1 \Rightarrow S = \overline{A} = 0$.

2.2.3.1 Tabela da Verdade da Função NÃO ou NOT

A tabela 2.5 apresenta casos possíveis da função NÃO.

A	S
0	1
1	0

Tabela 2.5

2.2.3.2 Inversor

O inversor é o bloco lógico que executa a função NÃO.

Suas representações simbólicas são vistas na figura 2.8.

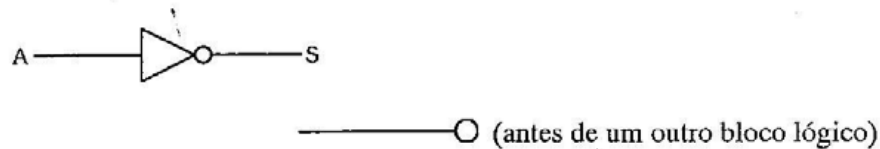



Figura 2.8

A função NÃO ou complementar também é conhecida como função NOT, termo derivado do inglês.

2.2.4 Função NÃO E, NE ou NAND

Como o próprio nome "NÃO E" diz: essa função é uma composição da função E com a função NÃO, ou seja, teremos a função E invertida. É representada algebricamente da seguinte forma:

$S = (\overline{A.B})$, onde o traço indica que temos a inversão do produto A.B.

2.2.4.1 Tabela da Verdade da Função NE ou NAND

A tabela 2.6 apresenta a função NE para 2 variáveis de entrada.

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela 2.6

Pela tabela da verdade, podemos notar que esta função é o inverso da função E.

2.2.4.2 Porta NE ou NAND

A porta NE é o bloco lógico que executa a função NE.

Sua representação simbólica é vista na figura 2.9.

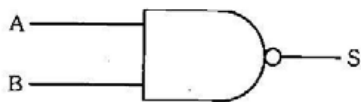



Figura 2.9

Podemos também formar uma porta NE através da composição de uma porta E com um inversor ligado a sua saída. A figura 2.10 mostra esta situação.

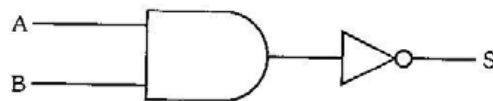



Figura 2.10

A porta NE, como outros blocos lógicos, pode ter 2 ou mais entradas. O termo NAND, também usual, é derivado do inglês.

2.2.5 Função NÃO OU, NOU ou NOR

Analogamente à função NE, a função NOU é a composição da função NÃO com a função OU, ou seja, a função NOU será o inverso da função OU. É representada da seguinte forma:

$S = (\overline{A + B})$, onde o traço indica a inversão da soma booleana A + B.

2.2.5.1 Tabela da Verdade da Função NOU ou NOR

A tabela 2.7 apresenta a função NOU para 2 variáveis de entrada.

A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Tabela 2.7

Podemos notar pela tabela da verdade que a função NOU representa a função OU invertida.

2.2.5.2 Porta NOU ou NOR

A porta NOU é o bloco lógico que executa a função NOU. Sua representação simbólica é vista na figura 2.11.

Figura 2.11

De maneira análoga, podemos formar uma porta NOU utilizando uma OU e um inversor ligado à sua saída. Esta situação é vista na figura 2.12

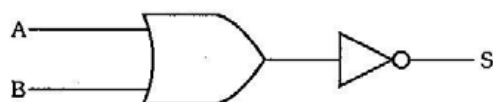



Figura 2.12

Podemos ter portas NOU com mais de 2 entradas. O termo NOR, também na prática utilizado, é derivado do inglês.

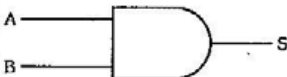
2.2.6 Quadro Resumo

BLOCOS LÓGICOS BÁSICOS				
Porta	**Símbolo Usual**	**Tabela da Verdade**	**Função Lógica**	**Expressão**
E **AND**	A, B, S	A B | S 0 0 | 0 0 1 | 0 1 0 | 0 1 1 | 1	**Função E:** assume 1 quando todas as variáveis forem 1 e 0 nos outros casos.	S = AB
OU **OR**	A, B, S	A B | S 0 0 | 0 0 1 | 1 1 0 | 1 1 1 | 1	**Função OU:** assume 0 quando todas as variáveis forem 0 e 1 nos outros casos.	S = A + B

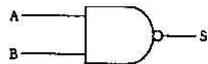
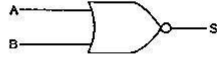
Tabela 2.8 (parte)

BLOCOS LÓGICOS BÁSICOS				
Porta	**Símbolo Usual**	**Tabela da Verdade**	**Função Lógica**	**Expressão**
NÃO NOT INVERSOR	A — S	A S 0 1 1 0	**Função NÃO:** inverte a variável aplicada à sua entrada.	$S=\overline{A}$
NE NAND	A, B — S	A B S 0 0 1 0 1 1 1 0 1 1 1 0	**Função NE:** inverso da função E.	$S=\overline{AB}$
NOU NOR	A, B — S	A B S 0 0 1 0 1 0 1 0 0 1 1 0	**Função NOU:** inverso da função OU.	$S=\overline{A+B}$

Tabela 2.8

2.3 Expressões Booleanas Obtidas de Circuitos Lógicos

Todo circuito lógico executa uma expressão booleana e, por mais complexo que seja, é formado pela interligação das portas lógicas básicas. Podemos obter a expressão booleana que é executada por um circuito lógico qualquer. Para mostrar o procedimento, vamos obter a expressão que o circuito da figura 2.13 executa:

Figura 2.13

Para facilitar, vamos dividir o circuito em 2 partes:

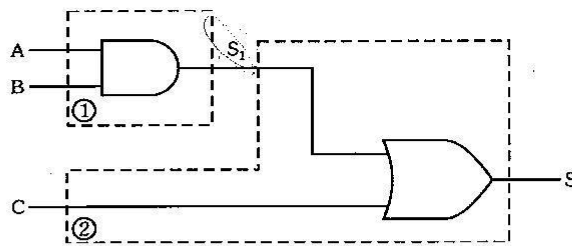



Figura 2.14

Na saída S_1, teremos o produto A.B, pois sendo este bloco uma porta E, sua expressão de saída será: S_1 = A.B. Como S_1 é injetada em uma das entradas da porta OU pertencente à segunda parte do circuito e na outra entrada está a variável C, a expressão de saída será: S = S_1 + C.

Para determinarmos a expressão final, basta agora, substituirmos a expressão de S_1 na expressão acima, obtendo então:

S = A . B + C

que é a expressão que o circuito executa.

Uma outra maneira mais simples para resolvermos o problema, é a de escrevermos nas saídas dos diversos blocos básicos do circuito, as expressões por estes executadas. A figura 2.15 ilustra este procedimento:

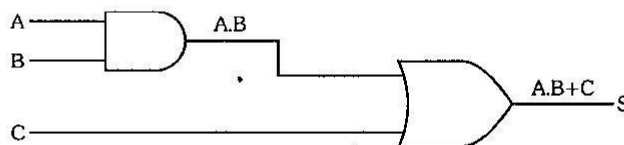



Figura 2.15

∴ S = A.B + C

2.3.1 Exercícios Resolvidos

1 - Escreva a expressão booleana executada pelo circuito da figura 2.16.

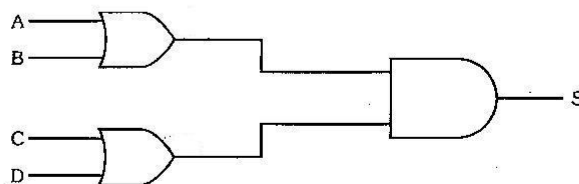



Figura 2.16

Vamos, agora, escrever as expressões de saída de cada bloco básico do circuito:

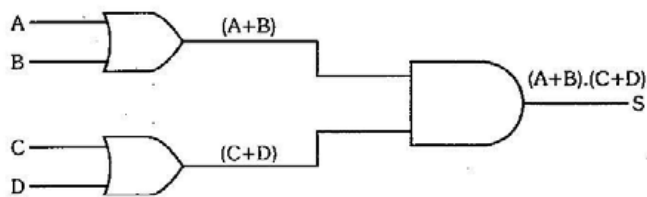



Figura 2.17

$\therefore S=(A+B) . (C+D)$

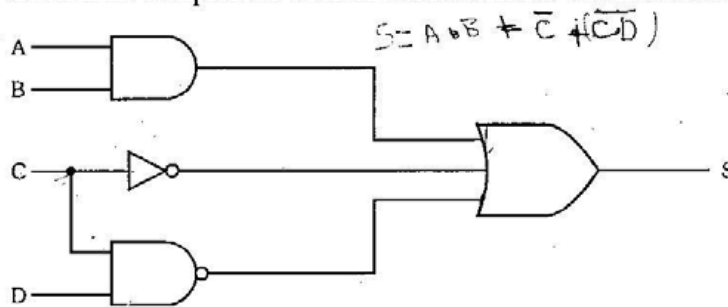
2 - Determine a expressão booleana característica do circuito da figura 2.18.

S= A.B + $\overline{C}$ + ($\overline{C.D}$)

Figura 2.18

Seguindo o processo descrito, temos:

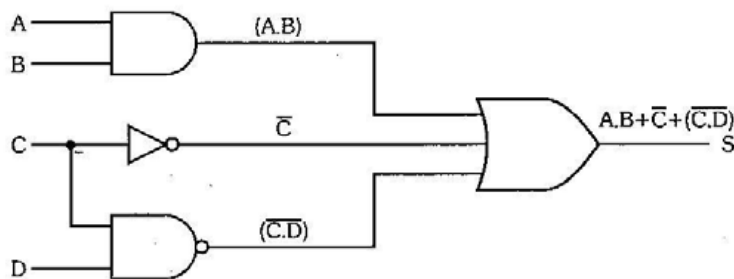



Figura 2.19

$\therefore S = A . B + \overline{C} + (\overline{C . D})$

3 - Idem, para o circuito da figura 2.18.

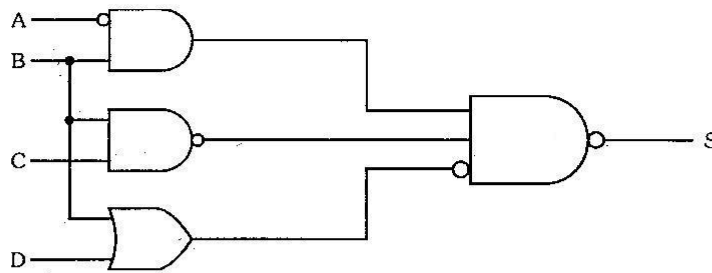



Figura 2.20

Antes da solução, convém recordarmos que os círculos colocados nas terminais de entrada junto às portas, representam também inversores. Solucionando, temos:

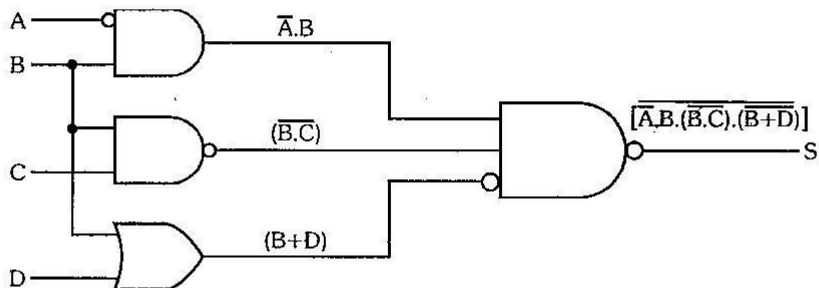



Figura 2.21

$$\therefore\ S = \left[\overline{\bar{A}\ .\ B\ .\ (\overline{B\ .\ C})\ .\ (\overline{B + D})}\right]$$

4 - Qual a expressão executada pelo circuito da figura 2.22?

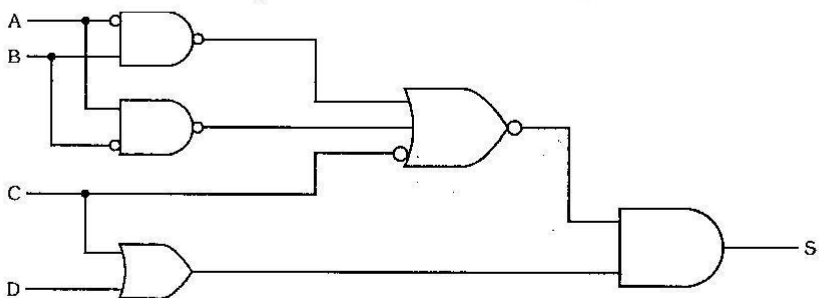



Figura 2.22

Resolução:

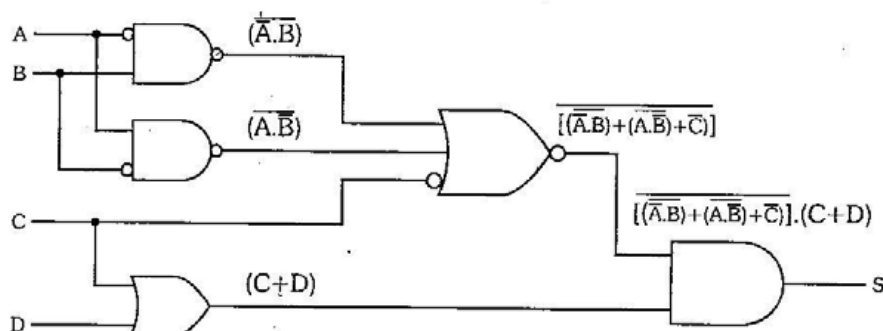



Figura 2.23

$$\therefore S = \left[\overline{(\overline{\bar{A}\ .\ B}) + (\overline{A\ .\ \bar{B}}) + \bar{C}}\right] . (C + D)$$

2.4 Circuitos Obtidos de Expressões Booleanas

Vimos, no tópico anterior, que podemos obter uma expressão booleana que um circuito lógico executa. Podemos também desenhar um circuito lógico que executa uma expressão booleana qualquer, ou seja, podemos desenhar um circuito a partir de sua expressão característica.

O método para a resolução consiste em se identificar as portas lógicas na expressão e desenhá-las com as respectivas ligações, a partir das variáveis de entrada. Para exemplificar, vamos obter o circuito que executa a expressão $S = (A + B) . C . (B + D)$.

Solucionaremos respeitando a hierarquia das funções da aritmética elementar, ou seja, iniciaremos a solução primeiramente pelos parênteses.

Para o primeiro parêntese, temos a soma booleana A + B, logo, o circuito que o executa será uma porta OU. Para o segundo, temos a soma booleana B + D, logo, o circuito será uma porta OU. Até aí, temos então:

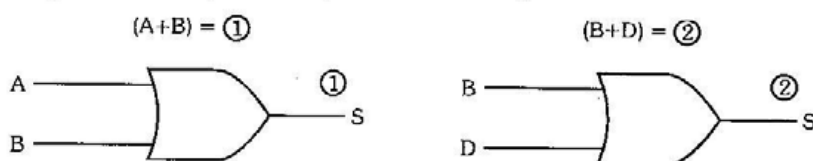



Figura 2.24

A seguir, temos uma multiplicação booleana dos dois parênteses, juntamente com a variável C, sendo o circuito que executa esta multiplicação, uma porta E:

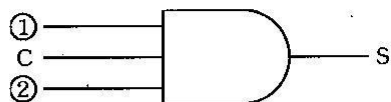



Figura 2.25

Substituindo as saídas ① e ② no bloco da figura 2.25, obtemos o circuito completo, que é visto na figura 2.26.

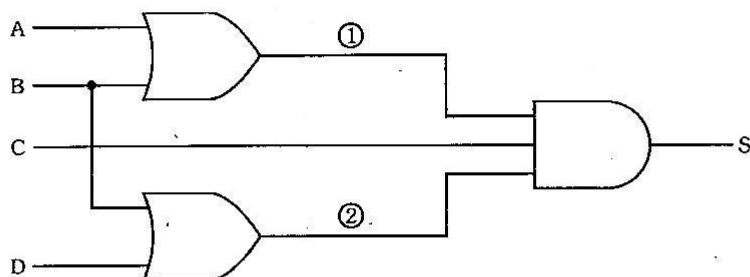



Figura 2.26

2.4.1 Exercícios Resolvidos

1- Desenhe o circuito que executa a expressão booleana $S = A.B.C + (A+B).C$.

Primeiramente, para identificar as portas lógicas, vamos numerar cada termo da expressão:

$$S = \underbrace{\underbrace{A.B.C}_{(1)} + \underbrace{\underbrace{(A+B)}_{(2)} . C}_{(3)}}_{(4)}$$

Assim sendo, temos:

① porta E com A, B e C.

② porta OU com A e B.

③ porta E com ② e C.

④ porta OU com ① e ③.

Para facilitar as ligações, pode-se utilizar uma rede ou barra de variáveis de entrada. A figura 2.27 mostra o circuito final, composto pela ligação das portas envolvidas com uma rede de variáveis de entrada.

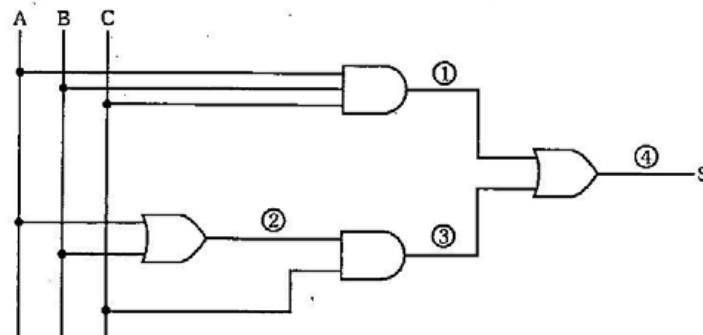



Figura 2.27

2 - Idem para a expressão: $S = [(\overline{\overline{A}+B}) + (\overline{\overline{C}.D})] . \overline{D}$.

Solucionando, conforme já explicado, temos:

$$S = \underbrace{\underbrace{[(\overline{\overline{A}+B})}_{(1)} + \underbrace{(\overline{\overline{C}.D})]}_{(2)}}_{(3)} \underbrace{. \overline{D}}_{}$$

$$S = \underbrace{\underbrace{[\underbrace{(\overline{\overline{A}+B})}_{(1)} + \underbrace{(\overline{\overline{C}.D})]}_{(2)}}_{(3)} . \overline{D}}_{(4)}$$

Desenhando e interligando as portas a partir de uma rede de variáveis de entrada, obtemos o circuito da figura 2.28.

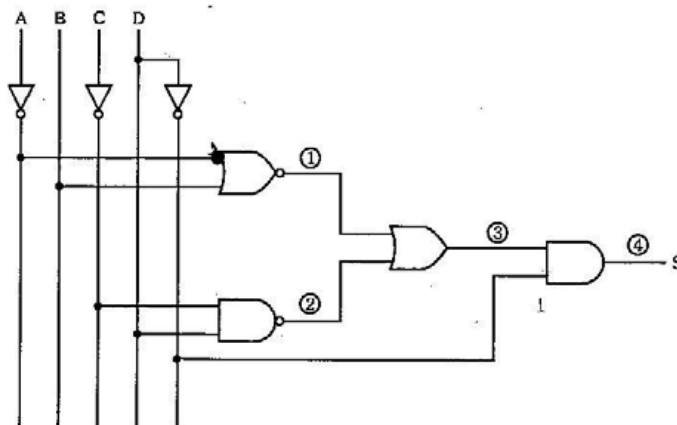



Figura 2.28

3 - Idem para a expressão: $S = \overline{[(\overline{\overline{A}.B}) + (\overline{C.\overline{D}})]}.E + \overline{A}.(A.\overline{D}.\overline{E} + C.D.E)$.

Solução:

$$S = \overline{[(\overline{\overline{A}.B}) + (\overline{C.\overline{D}})]}.E + \overline{A}.(A.\overline{D}.\overline{E} + C.D.E).$$

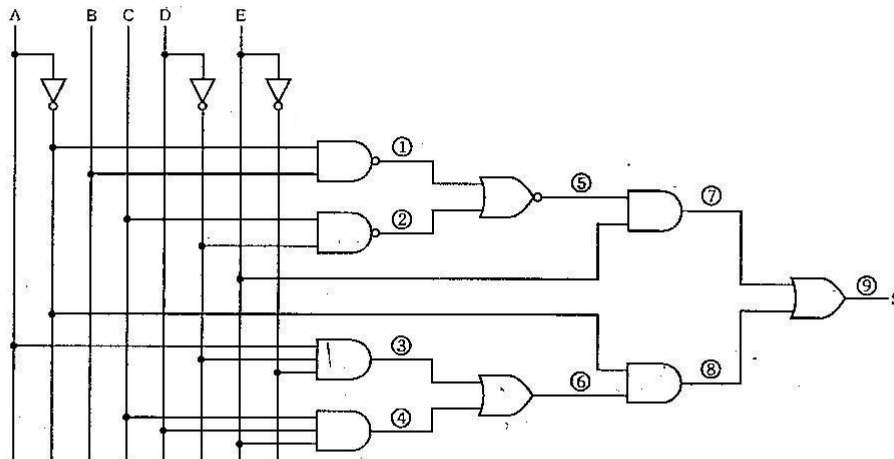



Figura 2.29

2.5 Tabelas da Verdade Obtidas de Expressões Booleanas

Uma maneira de se fazer o estudo de uma função booleana é a utilização da tabela da verdade, que, como vimos anteriormente, é um mapa onde se colocam todas as situações possíveis de uma dada expressão, juntamente com o valor por esta assumido.

Para extrairmos a tabela da verdade de uma expressão, acompanhamos o seguinte procedimento:

1º) Montamos o quadro de possibilidades.

2º) Montamos colunas para os vários membros da expressão.

3º) Preenchemos estas colunas com seus resultados.

4º) Montamos uma coluna para o resultado final.

5º) Preenchemos esta coluna com os resultados finais.

Para esclarecer este processo, vamos utilizar a expressão $S = A \cdot \overline{B} \cdot C + A \cdot \overline{D} + \overline{A} \cdot B \cdot D$.

Temos na expressão, 4 variáveis: A, B, C e D, logo, teremos 2^4 possibilidades de combinação de entrada.

Vamos, a seguir, montar o quadro de possibilidades com 4 variáveis de entrada, três colunas auxiliares, sendo uma para cada membro da expressão, e uma coluna para o resultado final (S):

A	B	C	D	1º membro $A \cdot \overline{B} \cdot C$	2º membro $A \cdot \overline{D}$	3º membro $\overline{A} \cdot B \cdot D$	Resultado final S
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	0	0	0	0
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	0	0	1	1
0	1	1	0	0	0	0	0
0	1	1	1	0	0	1	1
1	0	0	0	0	1	0	1
1	0	0	1	0	0	0	0
1	0	1	0	1	1	0	1
1	0	1	1	1	0	0	1
1	1	0	0	0	1	0	1
1	1	0	1	0	0	0	0
1	1	1	0	0	1	0	1
1	1	1	1	0	0	0	0

Tabela 2.9

Convém observar que na coluna relativa ao 1° membro, colocamos os resultados da multiplicação de A com o inverso da variável B e C $(A\ .\ \overline{B}\ .\ C)$. Na do 2° membro, o resultado de $A\overline{D}$ e na do 3°, o resultado de $\overline{A}\ .\ B\ .\ D$. Na coluna de resultado final (S) colocamos a soma booleana (porta OU) dos 3 termos da expressão.

Um outro modo de resolução, porém mais prático, consiste no preenchimento direto da coluna com o resultado final. Para isto, basta montar o quadro de possibilidades conforme o número de variáveis, reconhecer na expressão operações notáveis que permitem a conclusão do resultado final de imediato e, por exclusão, ir executando as operações até o preenchimento total da tabela.

Para melhor entendimento, vamos levantar a tabela da expressão $S = \overline{A} + B + A.\overline{B}.\overline{C}$, utilizando este modo mais prático.

Primeiramente, vamos montar o quadro de possibilidade para as 3 variáveis envolvidas na expressão:

A	B	C	S
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Tabela 2.10

Logo após, vamos preencher a tabela utilizando os casos notáveis, que permitem a conclusão do resultado final imediato:

① Nos casos onde A = 0 $(\overline{A} = 1)$, temos S = 1, pois, sendo $\overline{A}=1$, temos na expressão: $S = 1 + B + A.\overline{B}.\overline{C} = 1$ (qualquer que sejam os valores assumidos pela variável B ou pelo termo $A.\overline{B}.\overline{C}$).

② Nos casos remanescentes onde B =1, temos S = 1, pois da mesma forma que nos casos anteriores $S = \overline{A} + 1 + A.\overline{B}.\overline{C} = 1$.

③ O termo $A.\overline{B}.\overline{C}$ será igual a 1, somente no caso remanescente 100, levando para este caso a saída da expressão para 1 (S = 1).

④ Por exclusão, ou ainda, por substituição dos valores, concluímos que no último caso (101), temos na saída S = 0.

A tabela 2.11 apresenta o resultado com todos os casos preenchidos e assinalados conforme a análise efetuada.

A	B	C	S	
0	0	0	1	①
0	0	1	1	①
0	1	0	1	①
0	1	1	1	①
1	0	0	1	③
1	0	1	0	④
1	1	0	1	②
1	1	1	1	②

Tabela 2.11

2.5.1 Exercícios Resolvidos

1 - Prove as identidades abaixo relacionadas:

a) $\overline{A} . \overline{B} \neq \overline{A . B}$

b) $\overline{A} + \overline{B} \neq \overline{A + B}$

c) $\overline{A} . \overline{B} = \overline{A + B}$

d) $\overline{A} + \overline{B} = \overline{A . B}$

Podemos provar estas identidades, levantando as respectivas tabelas da verdade. Para facilitar, vamos colocar um quadro de possibilidades (2 variáveis: A e B) e colunas para os termos comuns entre as expressões:

A	B	$\overline{A}.\overline{B}$	$\overline{A.B}$	$\overline{A}+\overline{B}$	$\overline{A+B}$
0	0	1	1	1	1
0	1	0	1	1	0
1	0	0	1	1	0
1	1	0	0	0	0

Tabela 2.12

Como podemos notar pela tabela, os termos $\overline{A}.\overline{B}$ e $\overline{A.B}$ assumem resultados diferentes (a) para as mesmas possibilidades de entrada, o mesmo ocorrendo com $\overline{A}+\overline{B}$ e $\overline{A+B}$ (b), porém o termo $\overline{A}.\overline{B}$ assume resultado idêntico a $\overline{A+B}$ (c), o mesmo acontecendo entre $\overline{A}+\overline{B}$ e $\overline{A.B}$ (d).

2 - Levante a tabela da verdade da expressão:

$S = (A + B) . \overline{(B.C)}$.

Seguindo o processo já visto, temos:

① Pela análise do termo (A + B), concluímos que nos casos da tabela onde A = B = 0, temos S = 0, pois $S = (0 + 0) . \overline{(B.C)}$. Para qualquer outro caso remanescente, este termo será igual a 1, sendo necessária a análise de $\overline{(B.C)}$, para a conclusão dos outros resultados finais.

② Nos casos remanescentes da tabela, onde B = 0 ou C = 0, temos S = 1, pois $\overline{(0.C)}$ ou $\overline{(B.0)}$ são iguais a 1, que multiplicado por (A + B) = 1, resulta em S = 1.

③ Nos 2 últimos casos, onde B = C = 1, temos S = 0, pois $S = (A + B) . \overline{(1.1)} = 0$.

Passando os casos analisados para a tabela, temos:

A	B	C	S	
0	0	0	0	①
0	0	1	0	①
0	1	0	1	②
0	1	1	0	③
1	0	0	1	②
1	0	1	1	②
1	1	0	1	②
1	1	1	0	③

Tabela 2.13

3 - Monte a tabela da verdade da expressão:

$S = [\overline{(A+B) . C}] + [\overline{D . (B+C)}]$.

Solução:

① Nos casos onde $C = 0$, temos $S = 1$, pois o 1º termo da expressão será igual a 1: $S = \overline{[(A+B).0]} + ... = 1$.

② O mesmo ocorre nos casos remanescentes onde $D = 0$, pois $S = ... + [\overline{0.(B+C)}] = 1$.

③ Nos casos remanescentes onde $B = 1$, temos $S = 0$, pois estando B presente nos 2 termos, temos: $S = [\overline{(A+1)1}] + [\overline{1.(1+C)}] = 0 + 0 = 0$ (para estes casos restantes $C = 1$ e $D = 1$).

④ Sendo $B = 0$ para os casos restantes, onde $A = 0$, temos $S = 1$, pois $S = \overline{[(0+0).1]} + ... = 1$.

⑤ No último caso (1011), por simples substituição, concluímos que $S = 0$.

A tabela 2.14 mostra o resultado final com todos estes casos preenchidos e assinalados.

A	B	C	D	S	
0	0	0	0	1	①
0	0	0	1	1	①
0	0	1	0	1	②
0	0	1	1	1	④
0	1	0	0	1	①
0	1	0	1	1	①
0	1	1	0	1	
0	1	1	1	0	③
1	0	0	0	1	①
1	0	0	1	1	①
1	0	1	0	1	②
1	0	1	1	0	⑤
1	1	0	0	1	①
1	1	0	1	1	①
1	1	1	0	1	②
1	1	1	1	0	③

Tabela 2.14

4 - Analise o comportamento do circuito da figura 2.30.

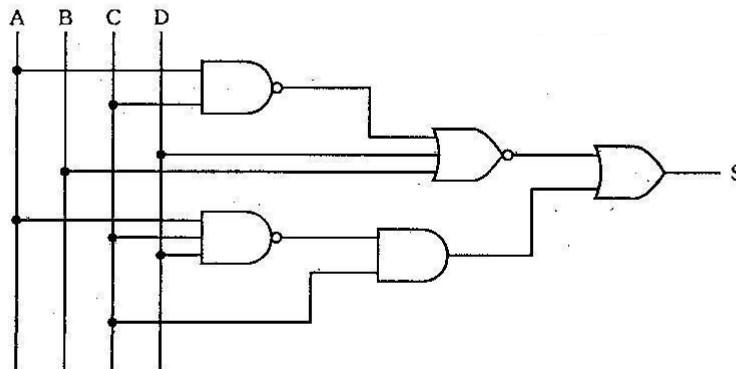



Figura 2.30

Para estudar o comportamento de um circuito lógico utilizamos sua tabela da verdade. Necessitamos, então, obter primeiramente a expressão a qual o circuito executa. Assim sendo, temos:

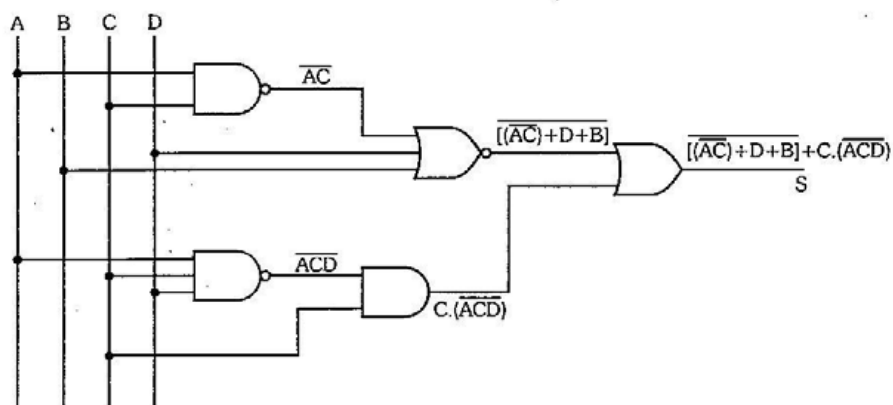



Figura 2.31

$\therefore S = \overline{[\overline{(A.C)} + D + B]} + C.\overline{(A.C.D)}$

Nesta expressão, para facilitar a obtenção do resultado final, vamos utilizar colunas auxiliares para obter os resultados relativos ao 1º e 2º termos. A tabela 2.15 apresenta a conclusão dos resultados.

A	B	C	D	$\overline{[\overline{(A.C)}+B+D]}$	$C.\overline{(ACD)}$	S
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	1	0	0	1	1
0	0	1	1	0	1	1
0	1	0	0	0	0	0
0	1	0	1	0	0	0
0	1	1	0	0	1	1
0	1	1	1	0	1	1
1	0	0	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	1	1
1	0	1	1	0	0	0
1	1	0	0	0	0	0
1	1	0	1	0	0	0
1	1	1	0	0	1	1
1	1	1	1	0	0	0

Tabela 2.15

2.6 Expressões Booleanas Obtidas de Tabelas da Verdade

Até aqui, vimos como obter expressões a partir de circuitos, circuitos a partir de expressões e tabelas da verdade a partir de circuitos ou expressões. Veremos, neste item, como podemos obter expressões e circuitos a partir de tabelas da verdade, sendo este o caso mais comum em projetos práticos, pois, geralmente, necessitamos representar situações através de tabelas da verdade e a partir destas, obter a expressão booleana e consequentemente, o circuito lógico.

Para demonstrar este procedimento, vamos obter a expressão da tabela 2.16.

A	B	S
0	0	1
0	1	0
1	0	1
1	1	1

Tabela 2.16

Observando a tabela, notamos que expressão é verdadeira (S = 1) nos casos onde A = 0 **e** B = 0 **ou** A = 1 **e** B = 0 **ou** A = 1 **e** B = 1. Para obter a expressão, basta montar os termos relativos aos casos onde a expressão for verdadeira e somá-los:

Caso 00: S = 1 quando, A = 0 e B = 0 ($\overline{A} = 1$ e $\overline{B} = 1$) $\Rightarrow \overline{A}.\overline{B}$

Caso 10: S = 1 quando, A = 1 e B = 0 (A = 1 e $\overline{B} = 1$) $\Rightarrow A.\overline{B}$

Caso 11: S = 1 quando, A = 1 e B = 1 $\Rightarrow$ A . B

$\therefore S = \overline{A}.\overline{B} + A.\overline{B} + A.B$

Notamos que o método permite obter, qualquer que seja a tabela, uma expressão padrão formada sempre pela soma de produtos. No próximo capítulo, relativo à álgebra de Boole, estudaremos o processo de simplificação desta, bem como, de outras expressões booleanas, possibilitando a obtenção de circuitos mais simplificados.

2.6.1 Exercícios Resolvidos

1 - Determine a expressão que executa a tabela 2.17 e desenhe o circuito lógico.

A	B	C	S
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Tabela 2.17

Para solucionar, extraímos os casos onde a expressão é verdadeira (S = 1): 000 ou 010 ou 110 ou 111.

$\therefore S = \overline{A}.\overline{B}.\overline{C} + \overline{A}.B.\overline{C} + A.B.\overline{C} + A.B.C$

Da expressão, extraímos o circuito lógico:

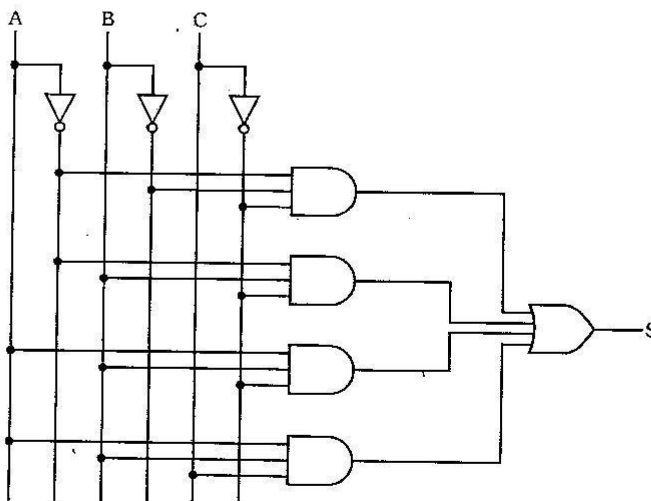



Figura 2.32

2 - Obtenha a expressão booleana da tabela 2.18.

A	B	C	D	S
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

Tabela 2.18

Da tabela extraímos os casos onde a expressão é verdadeira (S = 1):

0100 ou 0110 ou 1000 ou 1011.

$\therefore S = \overline{A}B\overline{C}\overline{D} + \overline{A}BC\overline{D} + A\overline{B}\overline{C}\overline{D} + A\overline{B}CD$

2.7 Blocos Lógicos OU EXCLUSIVO e COINCIDÊNCIA

Estudaremos, neste tópico, os blocos OU Exclusivo e Coincidência que são muito importantes na área de Eletrônica Digital, pois juntamente com as outras portas lógicas formam outros circuitos elementares dentro dos sistemas digitais.

Embora estes circuitos sejam blocos lógicos básicos, podemos considerá-los também como circuitos combinacionais, pois, como veremos em capítulos posteriores, sua obtenção provém de uma tabela da verdade (situação), que gera uma expressão característica, de onde esquematizamos o circuito.

2.7.1 Bloco OU EXCLUSIVO

A função que ele executa, como o próprio nome diz, consiste em fornecer 1 à saída quando as variáveis de entrada forem diferentes entre si. Com esta pequena apresentação podemos montar sua tabela da verdade e, obter pelo mesmo processo visto até aqui, sua expressão característica e, posteriormente, esquematizar o circuito:

A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 2.19

Da tabela obtemos sua expressão característica:

$S = \overline{A}.B + A.\overline{B}$.

Da expressão esquematizamos o circuito representativo da função OU Exclusivo, visto na figura 2.33.

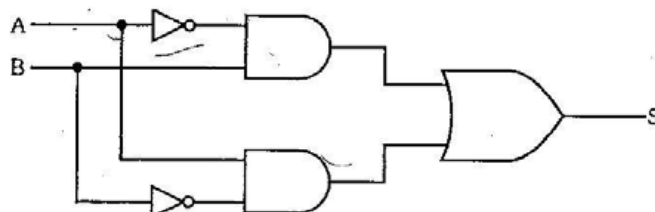



Figura 2.33

A notação algébrica que representa a função OU Exclusivo é $\mathbf{S = A \oplus B}$, onde se lê A OU Exclusivo B, sendo $S = A \oplus B = \overline{A}.B + A.\overline{B}$. O circuito OU Exclusivo pode ser representado tanto pelo circuito da figura 2.33, como também pelo símbolo visto na figura 2.34, sendo este mais simples.

Figura 2.34

Uma importante observação é que, ao contrário de outros blocos lógicos básicos, o circuito OU Exclusivo só pode ter 2 variáveis de entrada, fato este devido à sua definição básica. O circuito OU Exclusivo também é conhecido como Exclusive OR (EXOR), termo derivado do inglês.

2.7.2 Bloco COINCIDÊNCIA

A função que ele executa, como seu próprio nome diz, é a de fornecer 1 à saída quando houver uma coincidência nos valores das variáveis de entrada.

Vamos, agora, montar sua tabela da verdade:

A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

Tabela 2.20

A tabela gera a expressão $S = \overline{A}.\overline{B} + A.B$.

A partir desta expressão, vamos esquematizar o circuito:

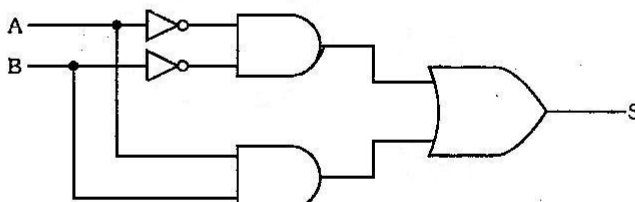



Figura 2.35

A notação algébrica que representa a função Coincidência é $\mathbf{S = A \odot B}$, onde se lê A Coincidência B, sendo $S = A \odot B = \overline{A}.\overline{B} + A.B$. O símbolo do circuito Coincidência é visto na figura 2.36.

Figura 2.36

Se compararmos as tabelas da verdade dos blocos OU Exclusivo e Coincidência, iremos concluir que estes são complementares, ou seja, teremos a saída de um invertida em relação à saída do outro. Assim sendo, podemos escrever:

$A \oplus B = \overline{A \odot B}$

O bloco Coincidência é também denominado de NOU Exclusivo e do inglês Exclusive NOR.

Da mesma forma que o OU Exclusivo, o bloco Coincidência é definido apenas para 2 variáveis de entrada.

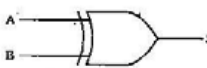
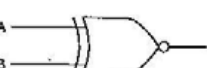
2.7.3 Quadro Resumo

BLOCOS LÓGICOS BÁSICOS				
Porta	**Símbolo Usual**	**Tabela da Verdade**	**Função Lógica**	**Expressão**
OU EXCLUSIVO **EXCLUSIVE OR**	A, B → S	A B S 0 0 0 0 1 1 1 0 1 1 1 0	**Função OU** Exclusivo: assume 1 quando as variáveis assumirem valores diferentes entre si.	$S=\overline{A}.B+A.\overline{B}$ $S=A\oplus B$
NOU EXCLUSIVO **EXCLUSIVE NOR** **COINCIDÊNCIA**	A, B → S	A B S 0 0 1 0 1 0 1 0 0 1 1 1	**Função Coincidência:** assume 1 quando houver coincidência entre os valores das variáveis.	$S=\overline{A}.\overline{B}+A.B$ $S=A\odot B$

Tabela 2.21

2.7.4 Exercícios Resolvidos

1 - A partir dos sinais aplicados às entradas da porta da figura 2.37, desenhe a forma de onda na saída S.

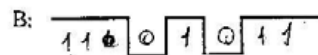
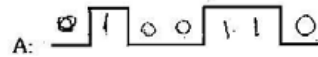
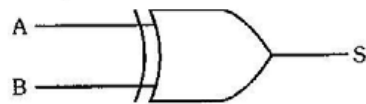



Figura 2.37

Para a solução, devemos desenhar o sinal de saída bit a bit, efetuando a operação OU Exclusivo entre os níveis, colocando S = 1 nos casos A = 0 e B = 1 ou A = 1 e B = 0. Assim sendo, temos:

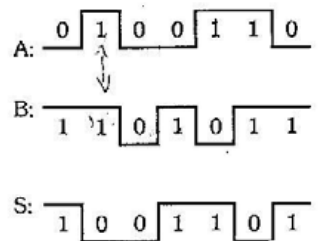



Figura 2.38

2 - Determine a expressão e a tabela da verdade do circuito visto na figura 2.39.

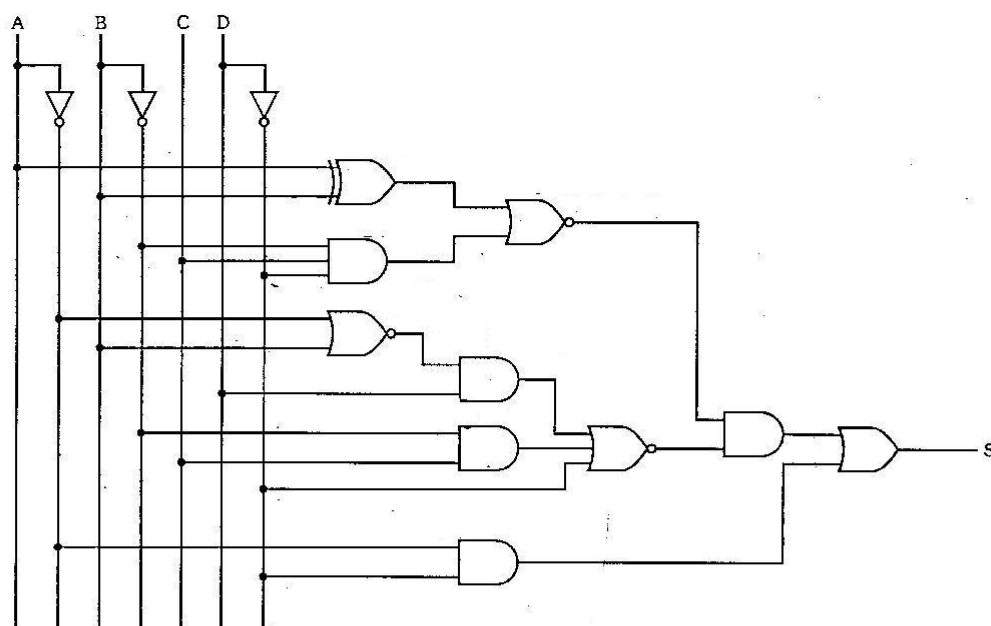



Figura 2.39

Vamos primeiramente obter a expressão que o circuito executa:

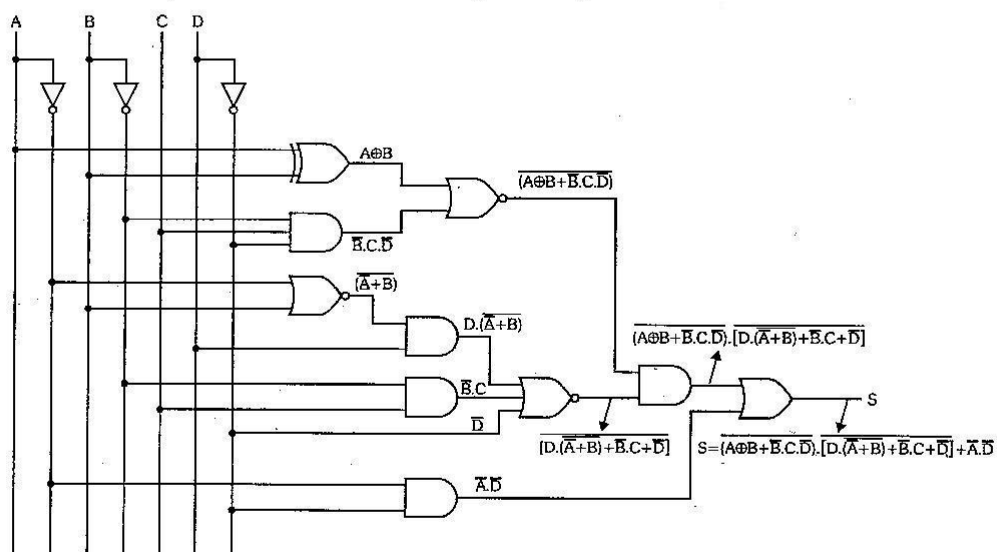



Figura 2.40

Logo após, a partir da expressão, vamos levantar sua tabela da verdade:

① Nos casos onde A = D = 0, temos S = 1, pois o termo $\overline{A}.\overline{D}$ será igual a 1: S = ... + 1 = 1.

② Nos casos remanescentes onde A = 1 e B = 0 ou A = 0 e B = 1 $(A \oplus B = 1)$, temos S = 0, pois o termo $\overline{(A \oplus B + \overline{B}C\overline{D})}$ será 0 e este multiplica o outro termo da expressão.

③ Nos casos remanescentes onde D = 0, temos S = 0, pois o termo $\overline{[\ldots + \overline{D}]}$ da mesma forma será 0.

④ Para o caso restante, onde $\overline{B}.C = 1$ (B = 0 e C = 1), da mesma forma S = 0.

⑤ Para os casos restantes onde B = 1, temos S = 1, pois o parêntese $\overline{(\overline{A}+B)}$ será 0, sendo $S = 1.\overline{[0+0+0]}+0=1$.

⑥ No último caso S = 1, pois sendo A = 0 valem as mesmas considerações do caso ⑤.

A tabela 2.22 mostra a tabela obtida da expressão, com os casos analisados assinalados.

A	B	C	D	S	
0	0	0	0	1	①
0	0	0	1	1	⑥
0	0	1	0	1	①
0	0	1	1	0	④
0	1	0	0	1	①
0	1	0	1	0	②
0	1	1	0	1	①
0	1	1	1	0	②
1	0	0	0	0	②
1	0	0	1	0	②
1	0	1	0	0	②
1	0	1	1	0	②
1	1	0	0	0	③
1	1	0	1	1	⑤
1	1	1	0	0	③ ⑤
1	1	1	1	1	⑤

Tabela 2.22

2.8 Equivalência entre Blocos Lógicos

Vamos estudar, neste tópico, como podemos obter circuitos equivalentes a inversores a partir das portas NE e NOU, e ainda, como formar portas NOU e OU utilizando portas NE , E e inversores, e por último, como obter portas NE e E utilizando portas OU, NOU e inversores.

Todas estas equivalências são muito importantes na prática, na montagem de sistemas digitais, pois possibilitam maior otimização na utilização dos circuitos integrados comerciais, assegurando principalmente a redução de componentes e a conseqüente minimização do custo dos sistemas.

2.8.1 Inversor a partir de uma Porta NE

Vamos analisar a tabela da verdade de uma porta NE:

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela 2.23

Podemos notar que no caso A = 0 e B = 0, a saída assume valor 1, e no caso A = 1 e B = 1, a saída assume valor 0.

Interligando os terminais de entrada da porta, estaremos fornecendo o mesmo nível às 2 entradas (A = B). Sendo este nível igual a 0 a saída é igual a 1, e sendo este nível igual a 1, a saída é 0, estando assim, formado um inversor. A figura 2.41 mostra uma porta NE com as entradas curto-circuitadas, formando um inversor, e a tabela 2.24 sua função lógica.

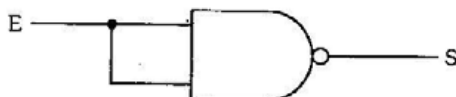



Figura 2.41

E	S
0	1
1	0

Tabela 2.24

Uma outra maneira de realizar a mesma equivalência consiste em fixar uma das entradas da porta NE no nível 1 e utilizar a outra como sendo a entrada do inversor. A observação dos 2 últimos casos da tabela 2.23 explica este modo de ligação (sendo A = 1: B = 0 ⇒ S = 1 e B = 1 ⇒ S = 0). A figura 2.42 ilustra o outro modo de se obter um inversor a partir de uma porta NE.

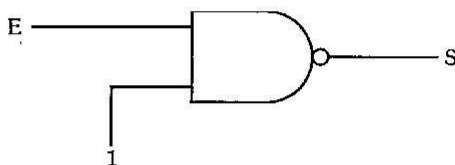



Figura 2.42

2.8.2 Inversor a partir de uma Porta NOU

Analogamente ao caso anterior, vamos analisar a tabela da verdade de uma porta NOU:

A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Tabela 2.25

Interligando A e B, cairemos num caso idêntico ao do item anterior, transformando a porta NOU em um inversor. A figura 2.43 e a tabela 2.25 mostram esta situação.

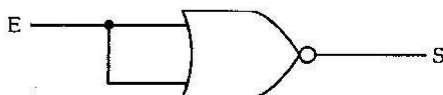



Figura 2.43

E	S
0	1
1	0

Tabela 2.26

Observando a tabela 2.25 nos dois primeiros casos, concluímos que uma outra maneira é a de aterrar (fornecer nível 0) uma das entradas e utilizar a outra como sendo a entrada do inversor (A = 0 : B = 0 ⇒ S = 1 e B = 1 ⇒ S = 0). A figura 2.44 apresenta esta outra maneira de se obter um inversor a partir de uma porta NOU.

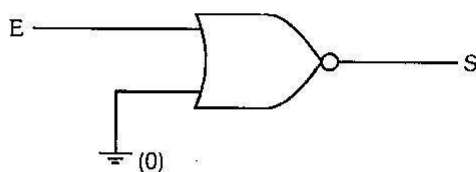



Figura 2.44

2.8.3 Portas NOU e OU a partir de E, NE e Inversores

Estas equivalências são obtidas a partir dos **Teoremas de De Morgan** a serem estudados no capítulo relativo à álgebra de Boole, sendo um deles a identidade $\overline{A+B} = \overline{A}.\overline{B}$, mostrando que uma porta NOU pode ser formada por um E com suas entradas invertidas. A figura 2.45 mostra esta equivalência e a tabela 2.27 prova a igualdade.

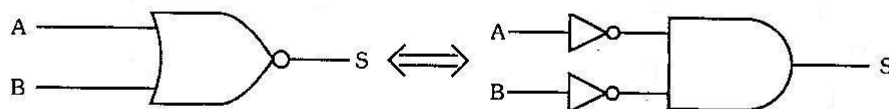



Figura 2.45

A	B	$\overline{A+B}$	$\overline{A}.\overline{B}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

Tabela 2.27

Colocando, agora, um inversor à saída de cada bloco da figura 2.45 obtemos a equivalência entre uma OU (obtida pela anulação mútua dos inversores) e uma NE com as 2 entradas invertidas, conforme mostra a figura 2.46.

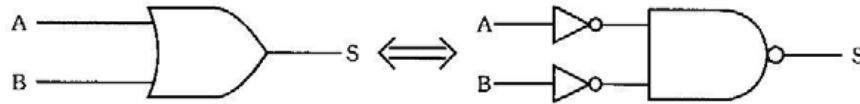



Figura 2.46

As equivalências podem ser estendidas para portas com mais de 2 variáveis de entrada.

2.8.4 Portas NE e E a partir de OU, NOU e Inversores

A partir da identidade $\overline{A.B} = \overline{A}+\overline{B}$ (outro teorema de De Morgan) obtemos a equivalência entre uma porta NE e a porta OU com suas entradas invertidas. A figura 2.47 mostra esta equivalência e a tabela 2.28 prova a igualdade.

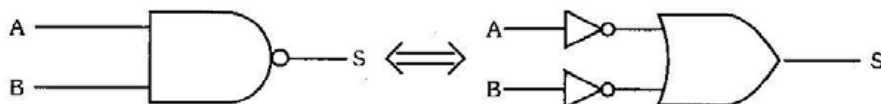



Figura 2.47

A	B	$\overline{A.B}$	$\overline{A}+\overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

Tabela 2.28

Da mesma forma que no caso anterior, colocando um inversor à saída de cada bloco, obtemos a equivalência entre uma E (obtida pela anulação mútua dos inversores) e uma NOU com suas entradas invertidas, conforme mostra a figura 2.48.

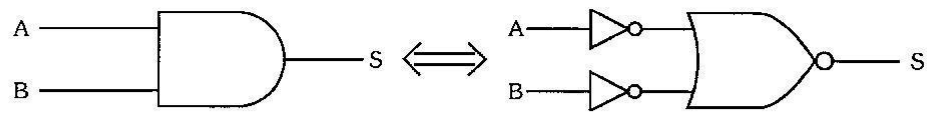



Figura 2.48

Também nestes casos, as equivalências podem ser estendidas para portas com mais de 2 variáveis de entrada.

2.8.5 Quadro Resumo

Tabela 2.29

2.8.6 Exercícios Resolvidos

1 - Desenhe o circuito OU Exclusivo, utilizando apenas portas NE.

Para a solução, vamos primeiramente desenhar o circuito na sua forma comum:

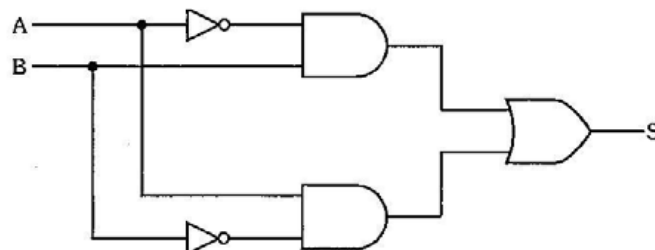



Figura 2.49

Em seguida, vamos efetuar um novo desenho, substituindo cada bloco lógico pelo equivalente composto apenas por portas NE. A figura 2.50 mostra o novo desenho, com as equivalências identificadas.

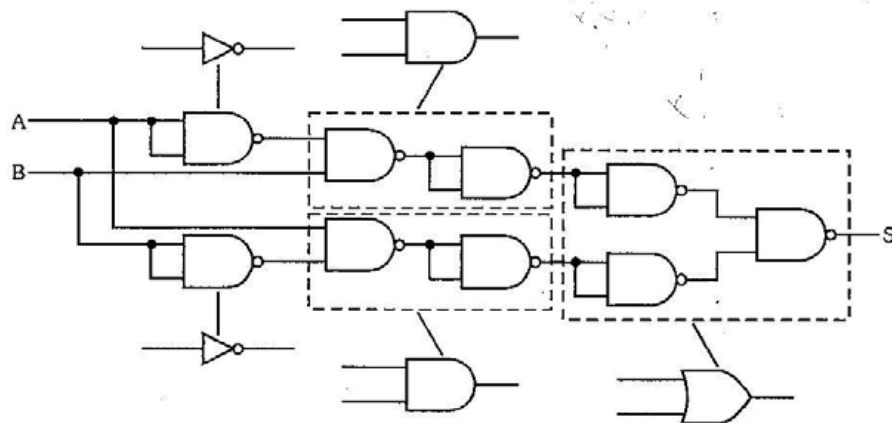



Figura 2.50

O circuito obtido pode ser ainda minimizado devido ao surgimento de inversores em série (saídas dos conjunto equivalentes a E com entradas do conjunto equivalente a OU). A figura 2.51 mostra o circuito final com a simplificação feita.

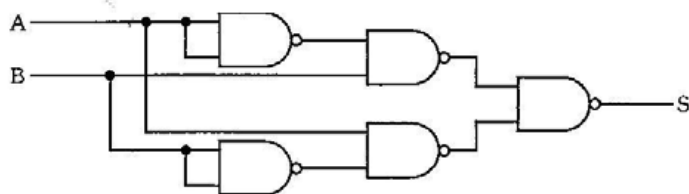



Figura 2.51

2 - Desenhe o circuito que executa a expressão somente com portas NOU:

$$S = A + (B \odot C)\ \overline{(\overline{A}.\overline{B}.C)} + \overline{(\overline{A}.\overline{C}+\overline{B})}\,.$$

A primeira etapa, que é o desenho do circuito a partir da expressão, de maneira convencional, é vista na figura 2.52.

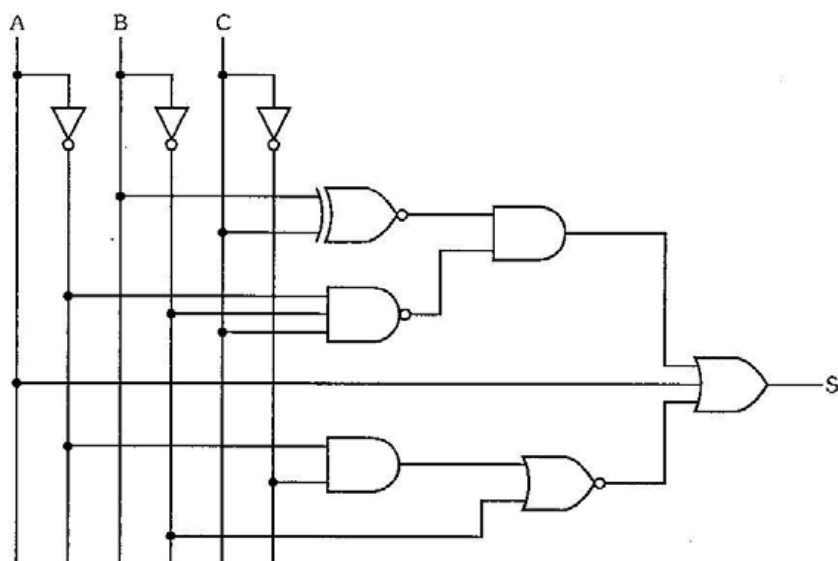



Figura 2.52

A segunda etapa é um novo desenho com os blocos substituídos pelos seus equivalentes compostos semente com NOU. Devemos lembrar que a equivalência vale, da mesma forma, para blocos lógicos básicos de mais entradas. Este circuito é visto na figura 2.53.

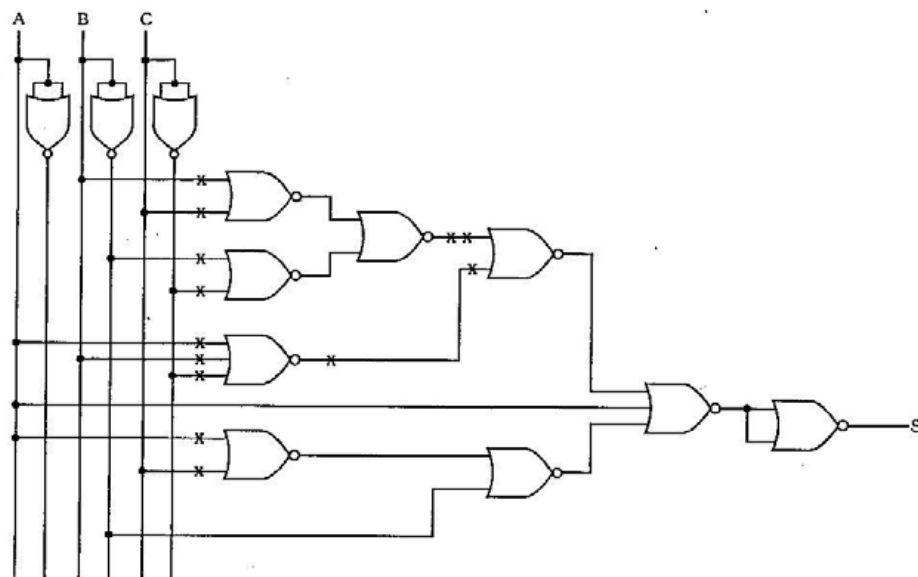



Figura 2.53

No circuito, assinalamos com x os inversores eliminados por estarem dispostos em série. Da mesma forma, foram eliminados e assinalados os inversores junto à rede de entrada, pela simples ligação no fio inverso da variável.

2.9 Exercícios Propostos

2.9.1 - De forma análoga aos circuitos das figuras 2.1, 2.4 e 2.7, esquematize os circuitos representativos das funções NE e NOU.

2.9.2 - Determine a expressão característica do circuito da figura 2.54.

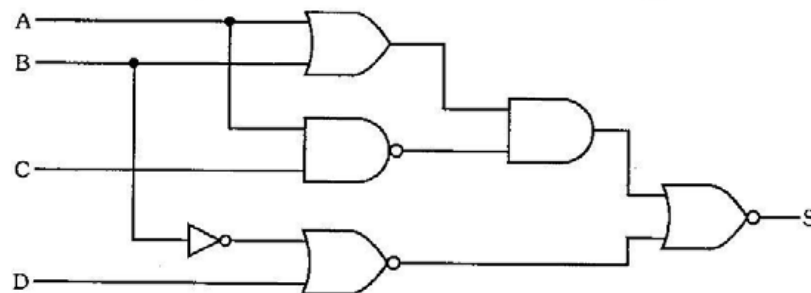



Figura 2.54

2.9.3 - Idem ao anterior, para o circuito da figura 2.55.

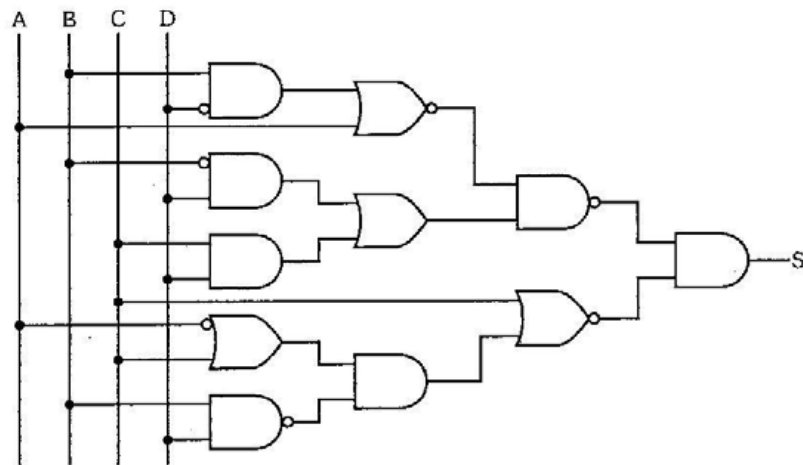



Figura 2.55

2.9.4 - Idem aos anteriores, para o circuito da figura 2.56.

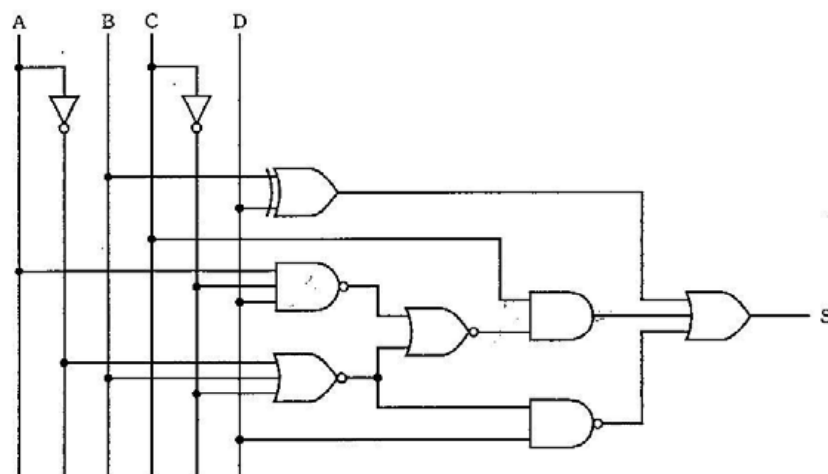



Figura 2.56

2.9.5 - Desenhe o circuito que executa a expressão:

$$S = \overline{A}.\overline{[\overline{B}.C + A.\overline{(C+\overline{D})} + B.\overline{C}.D]} + B.\overline{D}$$

2.9.6 - Idem ao anterior, para a expressão:

$$S = (A \odot B) . \overline{[A.\overline{B} + (\overline{\overline{B}} + \overline{\overline{D}}) + C.\overline{D} + \overline{(B.C)}]} + \overline{A}.B.\overline{C}.D$$

2.9.7 - Levante a tabela da verdade da expressão:

$$S = \overline{C}.\overline{[A.\overline{B} + B.(\overline{A} + C)]}$$

2.9.8 - Escreva a expressão característica do circuito da figura 2.57 e levante sua respectiva tabela da verdade.

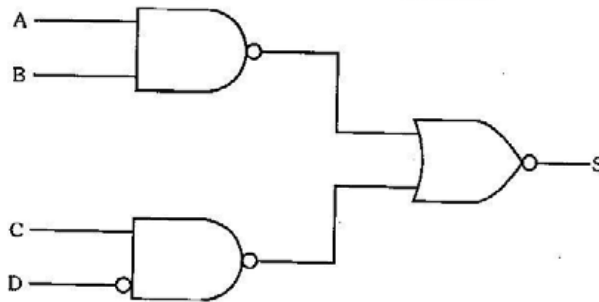



Figura 2.57

2.9.9 - Desenhe o circuito a partir da expressão e levante sua tabela da verdade:

$$S = \overline{[\overline{(\overline{B} + \overline{C} + \overline{D})} . \overline{(\overline{A} + B + C)} + C]} + A.\overline{B}.C + \overline{B}.\overline{(A + C)}$$

2.9.10 - Levante a tabela da verdade da expressão:

$$S = (B \oplus D) . \overline{[\overline{A} + \overline{B}.\overline{(C + \overline{D})} + A.\overline{B}.\overline{C}]}$$

2.9.11 - Prove que: $A \odot (B \oplus C) = A \oplus (B \odot C)$.

2.9.12 - Determine a expressão booleana a partir da tabela 2.30.

A	B	C	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1

Tabela 2.30 (parte)

A	B	C	S
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Tabela 2.30

2.9.13 - Desenhe o circuito que executa a tabela 2.31.

A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Tabela 2.31

2.9.14 - Desenhe o sinal na saída S do circuito da figura 2.58.

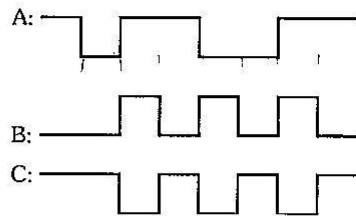
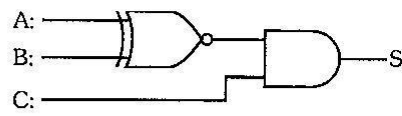



Figura 2.58

2.9.15 - Mostre que o circuito abaixo é um OU Exclusivo

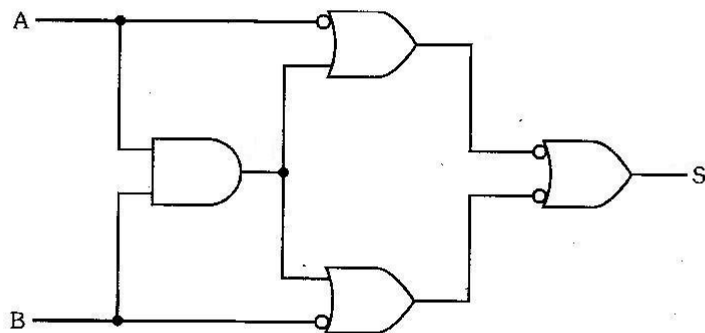



Figura 2.59

2.9.16 - Mostre que o circuito abaixo é um circuito coincidência.

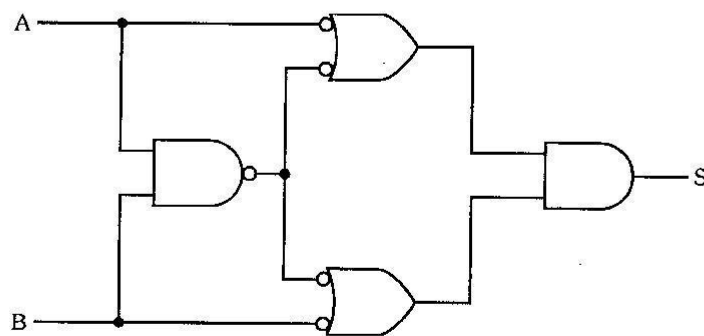



Figura 2.60

2.9.17 - Levante a tabela da verdade e esquematize o circuito que executa a seguinte expressão:

$$S = \{[A.B+C] \oplus [A+B]\} \odot C$$

2.9.18 - Esquematize o circuito coincidência, utilizando apenas portas NOU.

2.9.19 - Esquematize o circuito OU Exclusivo, utilizando somente 4 portas NE.

2.9.20 - Idem para o coincidência somente com 4 portas NOU.

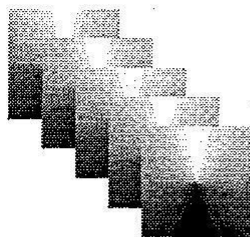
2.9.21 - Desenhe o circuito que executa a expressão do exercício 2.9.5 somente com portas NE.

2.9.22 - Idem para a expressão do 2.9.6, somente com portas NOU.

2.9.23 - Levante a tabela da verdade e, a partir desta, desenhe o circuito somente com portas NE:

$$S = (B \oplus C).\overline{[\overline{\overline{D}}+A.\overline{C}+D.(A+\overline{\overline{B}}+C)]}$$

2.9.24 - Esquematize o circuito da figura 2.56 (exercício 2.9.4) apenas com portas NOU.

CAPÍTULO 3

Álgebra de Boole e Simplificação de Circuitos Lógicos

3.1 Introdução

No capítulo anterior, trabalhamos com os circuitos lógicos sem nos preocuparmos com simplificações. Na prática, porém, estes circuitos obtidos admitem geralmente simplificações.

Para entrarmos no estudo da simplificação dos circuitos lógicos, teremos que fazer um breve estudo da Álgebra de Boole, pois é através de seus postulados, propriedades, teoremas fundamentais e identidades que efetuamos as mencionadas simplificações, e além disso, notamos que é na Álgebra de Boole que estão todos os fundamentos da Eletrônica Digital.

3.2 Variáveis e Expressões na Álgebra de Boole

Como vimos anteriormente, as variáveis booleanas são representadas através de letras, podendo assumir apenas dois valores distintos: 0 ou 1. Denominamos expressão booleana à sentença matemática composta de termos cujas variáveis são booleanas, da mesma forma, podendo assumir como resultado final 0 ou 1.

3.3 Postulados

A seguir, apresentaremos os postulados da complementação, da adição e da multiplicação da Álgebra de Boole, e suas respectivas identidades resultantes.

3.3.1 Postulados da Complementação

Este postulado, mostra como são as regras da complementação na álgebra de Boole. Chamaremos de $\overline{A}$ o complemento de A:

1º) Se $A = 0 \rightarrow \overline{A} = 1$

2º) Se $A = 1 \rightarrow \overline{A} = 0$

Através do postulado da complementação, podemos estabelecer a seguinte identidade:

$\overline{\overline{A}} = \mathbf{A}$

Se $A = 1$, temos: $\overline{A} = 0$ e se $\overline{A} = 0 \rightarrow \overline{\overline{A}} = 1$.

Se $A = 0$, temos: $\overline{A} = 1$ e se $\overline{A} = 1 \rightarrow \overline{\overline{A}} = 0$.

Assim sendo, podemos escrever: $\overline{\overline{A}} = A$.

O bloco lógico que executa o postulado da complementação é o Inversor.

3.3.2 Postulado da Adição

Este postulado, mostra como são as regras da adição dentro da Álgebra de Boole.

1º) $0 + 0 = 0$

2º) $0 + 1 = 1$

3º) $1 + 0 = 1$

4º) $1 + 1 = 1$

Através deste postulado, podemos estabelecer as seguintes identidades:

A+0 = A. A pode ser 0 ou 1, vejamos, então, todas as possibilidades:

$A= 0 \rightarrow 0 + 0 = 0$

$A= 1 \rightarrow 1 + 0 = 1$

Notamos que o resultado será sempre igual à variável A.

$A + 1 = 1$. Vejamos todas as possibilidades:

$A = 0 \rightarrow 0 + 1 = 1$

$A = 1 \rightarrow 1 + 1 = 1$

Notamos que se somarmos 1 a uma variável, o resultado será sempre 1.

$A + A = A$. Vejamos todas as possibilidades:

$A = 0 \rightarrow 0 + 0 = 0$

$A = 1 \rightarrow 1 + 1 = 1$

Notamos que se somarmos a mesma variável, o resultado será ela mesma.

$A + \overline{A} = 1$. Vejamos todas as possibilidades:

$A = 0 \rightarrow \overline{A} = 1 \rightarrow 0 + 1 = 1$

$A = 1 \rightarrow \overline{A} = 0 \rightarrow 1 + 0 = 1$

Notamos que sempre que somarmos a uma variável o seu complemento, teremos como resultado 1.

O bloco lógico que executa o postulado da adição é o OU.

3.3.3 Postulado da Multiplicação

É o postulado que determina as regras da multiplicação booleana:

1º) $0 . 0 = 0$

2º) $0 . 1 = 0$

3º) $1 . 0 = 0$

4º) $1 . 1 = 1$

Através deste postulado, podemos estabelecer as seguintes identidades:

$A . 0 = 0$. Podemos confirmar, verificando todas as possibilidades:

$A = 0 \rightarrow 0 . 0 = 0$

$A = 1 \rightarrow 1 . 0 = 0$

Notamos que todo número multiplicado por 0 é 0.

$A \cdot 1 = A$. Analisando todas as possibilidades, temos:

$A = 0 \rightarrow 0 \cdot 1 = 0$

$A = 1 \rightarrow 1 \cdot 1 = 1$

Notamos que o resultado destas expressões numéricas será sempre igual a A.

$A \cdot A = A$. Esta identidade, à primeira vista estranha, é verdadeira, como podemos confirmar pela análise de todas as possibilidades:

$A = 0 \rightarrow 0 \cdot 0 = 0$

$A = 1 \rightarrow 1 \cdot 1 = 1$

Notamos que os resultados serão sempre iguais a A.

$A \cdot \overline{A} = 0$. Vamos analisar todas as possibilidades:

$A = 0 \rightarrow 0 \cdot 1 = 0$

$A = 1 \rightarrow 1 \cdot 0 = 0$

Notamos que para ambos os valores possíveis que a variável pode assumir, o resultado da expressão será sempre 0.

O bloco lógico que executa o postulado da multiplicação é o E.

3.4 Propriedades

A seguir, descreveremos as principais propriedades algébricas, úteis principalmente, no manuseio e simplificação de expressões. Tal como na matemática comum, valem na Álgebra de Boole as propriedades comutativa, associativa e distributiva.

3.4.1 Propriedade Comutativa

Esta propriedade é válida tanto na adição, bem como na multiplicação:

Adição: A + B= B + A

Multiplicação: A . B = B . A

3.4.2 Propriedade Associativa

Da mesma forma que na anterior, temos a propriedade associativa válida na adição e na multiplicação:

Adição: A + (B + C) = (A + B) + C = A + B + C

Multiplicação: A . (B.C) = (A.B) . C = A.B.C

3.4.3 Propriedade Distributiva

A . (B+C) = A.B + A.C

Vamos verificar esta propriedade através da tabela verdade, analisando todas as possibilidades:

A	B	C	A(B+C)	AB + AC
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	0
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	1	1

Tabela 3.1

Notamos, pela tabela 3.1, que as expressões se equivalem.

3.5 Teoremas de De Morgan

Os teoremas de De Morgan são muito empregados na prática, em simplificações de expressões booleanas e, ainda, no desenvolvimento de circuitos digitais, como veremos em tópicos posteriores.

3.5.1 1º Teorema de De Morgan

O complemento do produto é igual à soma dos complementos:

$$\left(\overline{A \, . \, B}\right) = \overline{A} + \overline{B}$$

Para provar este teorema, vamos montar a tabela da verdade de cada membro e comparar os resultados:

A	B	$\overline{A \, . \, B}$	$\overline{A} + \overline{B}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

Tabela 3.2

Notamos a igualdade de ambas as colunas.

Este teorema foi aplicado no item referente à equivalência entre blocos lógicos (capítulo 2).

O teorema pode ser estendido para mais de duas variáveis:

$$\left(\overline{A.B.C \quad \dots \quad N}\right) = \overline{A} + \overline{B} + \overline{C} + \quad \dots \quad + \overline{N}$$

3.5.2 2º Teorema de De Morgan

O complemento da soma é igual ao produto dos complementos.

Este teorema é uma extensão do primeiro:

$$\left(\overline{A.B}\right) = \overline{A} + \overline{B} \quad \leftarrow 1^{\underline{o}} \text{ Teorema}$$

Podemos reescrevê-lo da seguinte maneira:

$$A \cdot B = \left(\overline{\overline{A} + \overline{B}}\right)$$

Notamos que A é o complemento de $\overline{A}$ e que B é o complemento de $\overline{B}$. Vamos chamar $\overline{A}$ de X e $\overline{B}$ de Y. Assim sendo, temos:

$$\overline{X} \cdot \overline{Y} = \left(\overline{X + Y}\right).$$

Reescrevendo, em termos de A e B, temos:

$$\overline{A} \cdot \overline{B} = \left(\overline{A + B}\right) \leftarrow 2^{\underline{o}} \text{ Teorema}$$

Da mesma forma que no anterior, o teorema pode ser estendido para mais de duas variáveis:

$$\left(\overline{A + B + C + \ldots + N}\right) = \overline{A} \cdot \overline{B} \cdot \overline{C} \ldots \overline{N}$$

Notamos, também, a aplicação deste teorema no item relativo à equivalência entre blocos lógicos.

3.6 Identidades Auxiliares

A seguir, vamos deduzir três identidades úteis para a simplificação de expressões.

3.6.1) A + A . B = A

Provamos esta identidade, utilizando a propriedade distributiva. Vamos evidenciar A no $1^{\underline{o}}$ termo:

$$A(1 + B) = A$$

Do postulado da soma temos: 1 + B = 1, logo podemos escrever:

$$A \cdot 1 = A \therefore A + AB = A$$

3.6.2) (A+B) . (A+C) = A + B.C

Vamos agora, provar esta identidade:

$(A+B).(A+C)$

$= A.A + A.C + A.B + B.C$ → Propriedade distributiva

$= A + A.C + A.B + B.C$ → Identidade $A.A = A$

$= A.(1 + B+C) + B.C$ → Propriedade distributiva

$= A . 1 + B.C$ → Identidades: $1+ X = 1$ e $A . 1= A$

$\therefore (A+B).(A+C) = A + BC$

3.6.3) $A + \overline{A}B = A + B$

Vamos, agora, provar esta identidade:

$A+\overline{A}.B = \left(\overline{\overline{A+\overline{A}.B}}\right)$ → Identidade $\overline{\overline{X}} = X$

$= [\overline{\overline{A} . (\overline{\overline{A}.B})}]$ → 2º Teorema de De Morgan: $\left(\overline{X + Y}\right) = \overline{X}.\overline{Y}$

$= [\overline{\overline{A} . (A+\overline{B})}]$ → 1º Teorema de De Morgan aplicado no parênteses: $\overline{X . Y} = \left(\overline{X}+\overline{Y}\right)$

$= (\overline{\overline{A} . A + \overline{A} . \overline{B}})$ → propriedade distributiva e identidade $\overline{A}.A = 0$

$= (\overline{\overline{A} . \overline{B}})$

$= (A + B)$ → 1º Teorema de De Morgan

$\therefore (A + \overline{A}.B) = A+B$

3.7 Quadro Resumo

POSTULADOS		
Complementação	**Adição**	**Multiplicação**
$A = 0 \rightarrow \overline{A} = 1$ $A = 1 \rightarrow \overline{A} = 0$	$0 + 0 = 0$ $0 + 1 = 1$ $1 + 0 = 1$ $1 + 1 = 1$	$0 . 0 = 0$ $0 . 1 = 0$ $1 . 0 = 0$ $1 . 1 = 1$
IDENTIDADES		
Complementação	**Adição**	**Multiplicação**
$\overline{\overline{A}} = A$	$A + 0 = A$ $A + 1 = 1$ $A + A = A$ $A + \overline{A} = 1$	$A . 0 = 0$ $A . 1 = A$ $A . A = A$ $A . \overline{A} = 0$
PROPRIEDADES		
Comutativa:	$A + B = B + A$ $A . B = B . A$	
Associativa:	$A + (B + C) = (A + B) + C = A + B + C$ $A . (B . C) = (A . B) . C = A . B . C$	
Distributiva:	$A . (B + C) = A . B + A . C$	
TEOREMAS DE MORGAN		
$\left(\overline{A . B}\right) = \overline{A} + \overline{B}$ $\left(\overline{A + B}\right) = \overline{A} . \overline{B}$		
IDENTIDADES AUXILIARES		
$A + A . B = A$ $A + \overline{A} . B = A + B$ $(A + B) . (A + C) = A + B . C$		

Tabela 3.3

3.8 Simplificação de Expressões Booleanas

Utilizando o conceito da Álgebra de Boole, podemos simplificar expressões e consequentemente circuitos.

Para efetuarmos estas simplificações, existem, basicamente, dois processos. O primeiro deles é a simplificação através da Álgebra de Boole, o segundo é a utilização dos mapas de Veitch-Karnaugh, como veremos no item 3.9. Para elucidar, vamos utilizar, por exemplo, a expressão:

$$S = ABC + A\overline{C} + A\overline{B}$$

Vamos simplificá-la, utilizando a Álgebra de Boole. Primeiramente, vamos evidenciar o termo A:

$$S = A\left(BC + \overline{C} + \overline{B}\right)$$

Agora, aplicando a propriedade associativa, temos:

$$S = A\left[BC + (\overline{C} + \overline{B})\right]$$

Aplicando a identidade $\overline{\overline{X}} = X$, temos:

$$S = A\left[BC + (\overline{\overline{\overline{C} + \overline{B}}})\right]$$

Aplicando o teorema de De Morgan, temos:

$$S = \left[BC + (\overline{BC})\right]A$$

Chamando BC de Y, logo $\left(\overline{BC}\right) = \overline{Y}$, temos então:

$$S = A(Y + \overline{Y})$$

Como $Y + \overline{Y} = 1$, logo: $S = A \, . \, 1 = A \; \therefore \; S = A$

Esta expressão mostra a importância da simplificação e a conseqüente minimização do circuito, pois os resultados são idênticos aos valores assumidos pela variável A, assim sendo, todo o circuito pode ser substituído por um único fio ligado à variável A.

Como um outro exemplo, vamos simplificar a expressão:

$$S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + A\overline{B}C$$

Tirando $\overline{A}\,.\,\overline{C}$ em evidência nos dois primeiros termos, temos:

$$S = \overline{A}.\overline{C}.(\overline{B} + B) + A\overline{B}C$$

Aplicando a identidade: $B + \overline{B} = 1$, temos:

$$S = \overline{A}.\overline{C}.(\overline{B} + B) + A\overline{B}C = \overline{A}.\overline{C} + A\overline{B}C \quad \therefore \quad S = \overline{A}\,\overline{C} + A\overline{B}C$$

3.8.1 Exercícios Resolvidos

1 - Simplifique as expressões booleanas, apresentadas a seguir

a) $S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}BC + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + AB\overline{C}$

Evidenciando $\overline{C}$, temos:

$$S = \overline{A}BC + \overline{C}(\overline{A}\,\overline{B} + \overline{A}B + A\overline{B} + AB)$$

Evidenciando $\overline{A}$ e A, temos:

$$S = \overline{A}BC + \overline{C}\left[\overline{A}(\overline{B} + B) + A(\overline{B} + B)\right]$$

$$S = \overline{A}BC + \overline{C}(\overline{A}.1 + A.1) \rightarrow \text{identidade } \overline{X} + X = 1$$

$$S = \overline{A}BC + \overline{C}(\overline{A} + A)$$

$$S = \overline{A}BC + \overline{C}.1 \rightarrow \text{identidade } \overline{X} + X = 1$$

$$S = \overline{\overline{(\overline{A}BC + \overline{C})}} \rightarrow \text{identidade } \overline{\overline{X}} = X$$

$$S = \overline{\left[\overline{(\overline{A}BC)}.C\right]} \rightarrow \text{teorema de De Morgan: } \left(\overline{X + Y}\right) = \overline{X}.\overline{Y}$$

$$S = \overline{\left[(A + \overline{B} + \overline{C}).C\right]} \rightarrow \text{teorema de De Morgan: } \left(\overline{X.Y.Z}\right) = \overline{X} + \overline{Y} + \overline{Z}$$

$$S = \overline{(AC + \overline{B}C + \overline{C}.C)} \rightarrow \text{propriedade distributiva}$$

$$S = \overline{\left[C.(A + \overline{B})\right]} \rightarrow \text{propriedade distributiva e identidade } X.\overline{X} = 0$$

$$S = \left[\overline{C} + \overline{(A + \overline{B})}\right] \rightarrow \text{teorema de De Morgan: } \left(\overline{X.Y}\right) = \overline{X} + \overline{Y}$$

$$S = (\overline{C} + \overline{A}.B) \rightarrow \text{teorema de De Morgan: } \left(\overline{X + Y}\right) = \overline{X}.\overline{Y}$$

$$\therefore S = \overline{C} + \overline{A}.B$$

b) $S = (A + B + C).(\overline{A} + \overline{B} + C)$

Aplicando a propriedade distributiva, temos:

$S = A\overline{A} + A\overline{B} + AC + \overline{A}B + B\overline{B} + BC + \overline{A}C + \overline{B}C + CC$

Vamos usar as identidades $X.\overline{X} = 0$ e X.X=X e reescrever:

$S = A\overline{B} + AC + \overline{A}B + BC + \overline{A}C + \overline{B}C + C$

Colocando C em evidência, temos:

$S = A\overline{B} + C(A + B + \overline{A} + \overline{B} + 1) + \overline{A}B$

Usando as identidades: X+1=1 e X.1=X, obtemos o resultado final:

$S = A\overline{B} + \overline{A}B + C$

c) $S = \left[\overline{(\overline{AC}) + B + D}\right] + C(\overline{ACD})$

Aplicando o teorema de De Morgan ao 1º e 2º termos, obtemos:

$S = (\overline{\overline{A} + \overline{C} + B + D}) + C(\overline{A} + \overline{C} + \overline{D})$

Agora, aplicando o teorema de De Morgan ao 1º termo e a propriedade distributiva ao 2º termo, temos:

$S = AC\overline{D}\,\overline{B} + \overline{A}C + C\overline{C} + C\overline{D}$

Reescrevendo, aplicando a identidade $X.\overline{X} = 0$, temos:

$S = A\overline{B}C\overline{D} + \overline{A}C + C\overline{D}$

Evidenciando o termo $C\overline{D}$, vamos ter:

$S = C\overline{D}(A\overline{B} + 1) + \overline{A}C$

$S = C\overline{D}.1 + \overline{A}C \rightarrow$ identidade X+1=1

$\therefore S = C\overline{D} + \overline{A}C$

2 - A partir da expressão $S = (\overline{A \odot B})$, obtenha $S = A \oplus B$.

O primeiro passo é substituir a expressão do circuito coincidência pela sua equivalente:

$S = (\overline{A \odot B})$

$S = (\overline{\overline{A}\,\overline{B} + AB})$

Aplicando De Morgan $\left(\overline{X+Y}\right)=\overline{X}.\overline{Y}$, temos:

$$S=\overline{\left(\overline{A}\,\overline{B}\right)}.\overline{\left(AB\right)}$$

Aplicando o outro teorema, $\left(\overline{X.Y}\right)=\overline{X}+\overline{Y}$, em cada parênteses, temos:

$$S=(A+B).(\overline{A}+\overline{B})$$

Aplicando a propriedade distributiva, temos:

$$S=A\overline{A}+\overline{A}B+A\overline{B}+\overline{B}B$$

Como $A\overline{A}=0$ e $\overline{B}B=0$, temos:

$$S=\overline{A}B+A\overline{B}\Rightarrow S=A\oplus B$$

3 - Obtenha o circuito simplificado que executa a expressão:

$$S=(A\oplus B)\left[\overline{B(A+\overline{C})+\overline{D}\overline{(\overline{A}+B+\overline{C})}}\right]$$

Aplicando a propriedade distributiva e De Morgan respectivamente aos termos do colchete, temos:

$$S=(A\oplus B)(\overline{AB+B\overline{C}+\overline{D}A\overline{B}C})$$

Reescrevendo o último termo, em ordem, temos:

$$S=(A\oplus B)\ (\overline{AB\ +\ B\overline{C}\ +\ A\overline{B}C\overline{D}})$$

Aplicando De Morgan ao 2º parêntese, temos:

$$S=(A\oplus B)\ (\overline{AB})\ (\overline{B\overline{C}})\ (\overline{A\overline{B}C\overline{D}})$$

Aplicando novamente em cada parêntese e substituindo o 1º pela expressão equivalente do OU Exclusivo, obtemos:

$$S=(\overline{A}B+A\overline{B})\ (\overline{A}+\overline{B})\ (\overline{B}+C)\ (\overline{A}+B+\overline{C}+D)$$

Efetuando a multiplicação entre os dois primeiros parênteses, eliminando os termos resultantes, onde: $\overline{A}\ .\ A=0$ e $\overline{B}\ .\ B=0$, obtemos:

$$S=(\overline{A}\,\overline{A}B+A\overline{B}\,\overline{B})\ (\overline{B}+C)\ (\overline{A}+B+\overline{C}+D)$$

Como x . x = x, temos:

$$S = (\overline{A}B + A\overline{B})(\overline{B} + C)(\overline{A} + B + \overline{C} + D)$$

Da mesma forma, efetuando a multiplicação entre os dois últimos, obtemos:

$$S = (\overline{A}B + A\overline{B})(\overline{A}\,\overline{B} + \overline{B}\,\overline{C} + \overline{B}D + \overline{A}C + BC + CD)$$

Novamente multiplicando, temos:

$$S = \overline{A}BC + \overline{A}BC + \overline{A}BCD + A\overline{B}\,\overline{C} + A\overline{B}D + A\overline{B}CD$$

Tirando em evidência $\overline{A}BC$ para os três primeiros termos e $A\overline{B}$ para os últimos, temos:

$$S = \overline{A}BC\,(1 + 1 + D) + A\overline{B}\,(\overline{C} + D + CD)$$

Fazendo (1 + D) = 1 e colocando em evidência D no 2º parêntese temos:

$$S = \overline{A}BC + A\overline{B}\,[\overline{C} + D(1 + C)]$$

Como 1 + C = 1, temos:

$$S = \overline{A}BC + A\overline{B}(\overline{C} + D)$$

$$\therefore\ S = \overline{A}BC + A\overline{B}\,\overline{C} + A\overline{B}D$$

A partir da expressão, desenhamos o circuito simplificado visto na figura 3.1.

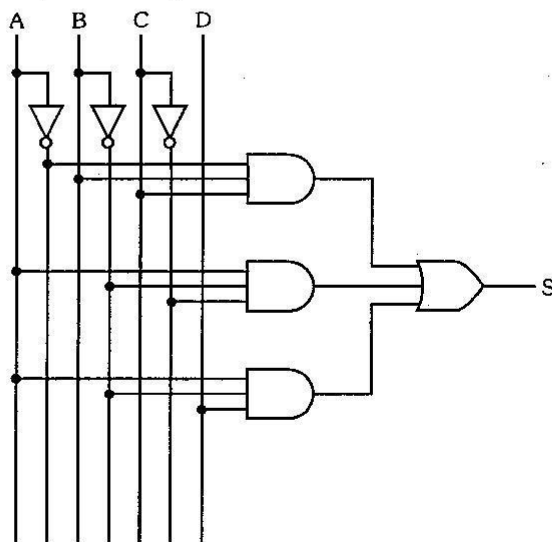



Figura 3.1

3.9 Simplificação de Expressões Booleanas através dos Diagramas de Veitch-Karnaugh

Vimos até aqui, a simplificação de expressões mediante a utilização dos postulados, propriedades e identidades da Álgebra de Boole. Nestes itens, vamos tratar da simplificação de expressões por meio dos diagramas de Veitch-Karnaugh.

Estes mapas ou diagramas permitem a simplificação de maneira mais rápida dos casos extraídos de tabelas da verdade, obtidas de situações quaisquer. Serão estudados os diagramas para 2, 3, 4 e 5 variáveis.

3.9.1 Diagrama de Veitch-Karnaugh para 2 Variáveis

A figura 3.2 mostra um diagrama de Veitch-Karnaugh para 2 variáveis:

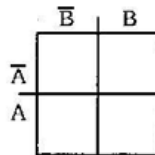



Figura 3.2

No mapa, encontramos todas as possibilidades assumidas entre as variáveis A e B. A figura 3.3 mostra todas as regiões do mapa.

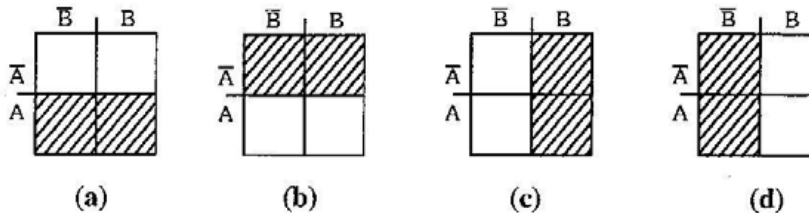



Figura 3.3 - Regiões do mapa de Veitch-Karnaugh:

(a) região onde A = 1.

(b) região onde $A = 0\ (\overline{A} = 1)$.

(c) região onde B = 1.

(d) região onde $B = 0\ (\overline{B} = 1)$.

Com 2 variáveis, podemos obter 4 possibilidades:

A	B	
0	0	→ caso 0
0	1	→ caso 1
1	0	→ caso 2
1	1	→ caso 3

Tabela 3.4

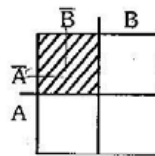
No caso 0, temos: A = 0 e B = 0. A região do diagrama que mostra esta condição é a da intersecção das regiões onde A = 0 e B = 0:

→ Esta região também pode ser chamada de região $\overline{A}\,\overline{B}$.

Figura 3.4

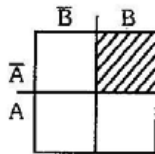
No caso 1, temos: A = 0 e B = 1. A região do diagrama que mostra esta condição é a da intersecção das regiões onde A = 0 ($\overline{A}$ = 1) e B = 1.

→ Esta região também pode ser chamada de região $\overline{A}B$.

Figura 3.5

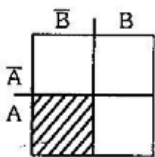
No caso 2, temos a intersecção das regiões onde A = 1 e B = 0 ($\overline{B}$ = 1). Fazendo esta intersecção, temos:

→ Esta região também pode ser chamada de região $A\overline{B}$

Figura 3.6

No caso 3, temos a intersecção das regiões onde A = 1 e B = 1. Fazendo esta intersecção, temos:

→ Esta região também pode ser chamada de região AB.

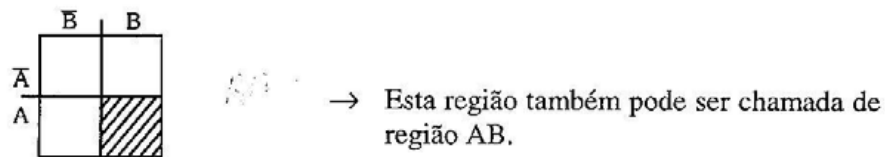



Figura 3.7

Podemos distribuir, então, as 4 possibilidades neste diagrama, da seguinte forma:

Figura 3.8

Logo, notamos que cada linha da tabela da verdade possui sua região própria no diagrama de Veitch-Karnaugh.

Essas regiões são, portanto, os locais onde devem ser colocados os valores que a expressão assume nas diferentes possibilidades .

Para entendermos melhor o significado destes conceitos vamos utilizar os exemplos:

1- A tabela da verdade mostra o estudo de uma função de 2 variáveis. Vamos colocar seus resultados no Diagrama de Veitch-Karnaugh.

A	B	S	
0	0	0	→ caso 0
0	1	1	→ caso 1
1	0	1	→ caso 2
1	1	1	→ caso 3

Tabela 3.5

Utilizando o método desenvolvido no capítulo 2, obtemos a expressão característica da função:

$S = \overline{A}B + A\overline{B} + AB$

Passando para o mapa os casos da tabela da verdade, conforme o esquema de colocação visto na figura 3.8, temos:

	$\overline{B}$	B
$\overline{A}$	0	1
A	1	1

Figura 3.9

Uma vez entendida a colocação dos valores assumidos pela expressão em cada caso no diagrama de Veitch-Karnaugh, vamos verificar como podemos efetuar as simplificações.

Para obtermos a expressão simplificada do diagrama, utilizamos o seguinte método:

Tentamos agrupar as regiões onde S é igual a 1, no menor número possível de agrupamentos. As regiões onde S é 1, que não puderem ser agrupadas, serão consideradas isoladamente. Para um diagrama de 2 variáveis, os agrupamentos possíveis são os seguintes:

a) Quadra:

Conjunto de 4 regiões, onde S é igual a 1. No diagrama de 2 variáveis, é o agrupamento máximo, proveniente de uma tabela onde todos os casos valem 1. Assim sendo, a expressão final simplificada obtida é $S = 1$. A figura 3.10 ilustra esta situação:

Figura 3.10

b) Pares:

Conjunto de 2 regiões onde S é 1, que tem um lado em comum, ou seja, são vizinhos. As figuras 3.11 e 3.12 mostram exemplos de 2 pares agrupados e suas respectivas expressões, dentre os 4 possíveis em 2 variáveis:

Figura 3.11

Figura 3.12

c) Termos isolados:

Regiões onde S é 1, sem vizinhança para grupamentos. São os próprios casos de entrada, sem simplificação. A figura 3.13 exemplifica 2 termos isolados, sem possibilidade de agrupamento.

Figura 3.13

Para o exemplo, efetuando os agrupamentos, temos:

Figura 3.14

Feito isto, escrevemos a expressão de cada par, ou seja, a região que o par ocupa no diagrama.

O par 1 ocupa a região onde A é igual a 1, então, sua expressão será: Par 1 = A.

O par 2 ocupa a região onde B é igual a 1, então, sua expressão será: Par 2 = B.

Notamos também que nenhum 1 ficou fora dos agrupamentos, e ainda que o mesmo 1 pode pertencer a mais de um agrupamento.

Para obter a expressão simplificada, basta, agora, somarmos os termos obtidos nos agrupamentos:

$$S = \text{Par } 1 + \text{Par } 2 \quad \therefore \quad S = A + B$$

Como podemos notar, esta é a expressão de uma porta OU, pois a tabela da verdade também é a da porta OU. Outro fato a ser notado é que a expressão obtida é visivelmente menor do que a extraída diretamente da tabela da verdade, acarretando um circuito mais simples, diminuindo, conseqüentemente, a dificuldade de montagem e o custo do sistema.

2 - Vamos simplificar o circuito que executa a tabela da verdade a seguir:

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela 3.6

Obtendo a expressão diretamente da tabela, temos:

$$S = \overline{A}\,\overline{B} + \overline{A}B + A\overline{B}$$

Transportando a tabela para o diagrama, mediante processo já visto, temos:

	$\overline{B}$	B
$\overline{A}$	1	1
A	1	0

Figura 3.15

Agora, vamos agrupar os pares:

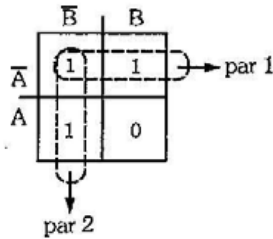



Figura 3.16

Vamos escrever as expressões dos pares:

par 1 -> $\overline{A}$

par 2 -> $\overline{B}$

Somando as expressões dos pares, temos a expressão simplificada:

$$S = \overline{A} + \overline{B}$$

Notamos que a tabela da verdade é a de uma porta NE. Aplicando o teorema de De Morgan à expressão, após a simplificação, encontramos a expressão de uma porta NE:

$$S = \overline{AB}$$

3.9.2 Diagramas de Veitch-Karnaugh para 3 Variáveis

O diagrama de Veitch-Karnaugh para 3 variáveis é visto na figura 3.17.

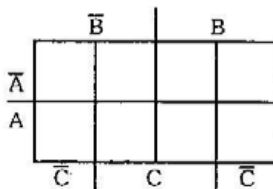



Figura 3.17

No mapa, encontramos todas as possibilidades assumidas entre as variáveis A, B e C. A figura 3.18 mostra as regiões deste mapa.

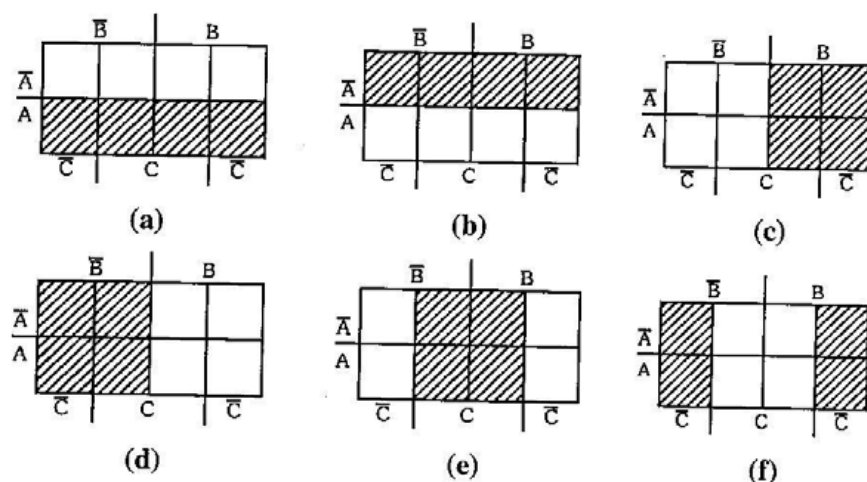



Figura 3.18 - Regiões do mapa de Veitch-Karnaugh:

(a) Região na qual A = 1.

(b) Região na qual $\overline{A} = 1(A = 0)$.

(c) Região na qual B = 1.

(d) Região na qual $\overline{B} = 1(B = 0)$.

(e) Região na qual C = 1.

(f) Região na qual $\overline{C} = 1(C = 0)$.

Neste diagrama, também temos uma região para cada caso da tabela da verdade. A tabela 3.7 e a figura 3.19 mostram os casos para 3 variáveis e as respectivas localizações no mapa.

Caso	A	B	C
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Tabela 3.7

	$\overline{B}$		B	
$\overline{A}$	Caso 0 000 $\overline{A}\overline{B}\overline{C}$	Caso 1 001 $\overline{A}\overline{B}C$	Caso 3 011 $\overline{A}BC$	Caso 2 010 $\overline{A}B\overline{C}$
A	Caso 4 100 $A\overline{B}\overline{C}$	Caso 5 101 $A\overline{B}C$	Caso 7 111 ABC	Caso 6 110 $AB\overline{C}$
	$\overline{C}$	C		$\overline{C}$

Figura 3.19

Vamos analisar a localização somente de uma das possibilidades, visto que as outras são de maneira análoga. Assim sendo, vamos localizar no diagrama o caso 3:

Caso	**A**	**B**	**C**
3	0	1	1

No diagrama, será a intersecção das regiões que: $A = 0(\overline{A} = 1)$, $B = 1$ e $C = 1$. Esta pode ser chamada de região $\overline{A}BC$. A figura 3.20 mostra esta localização no diagrama, para a colocação do respectivo caso de entrada da coluna S.

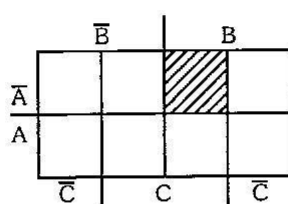



Figura 3.20

Para melhor compreensão, vamos, como exemplo, transpor para o diagrama as situações de saída da tabela 3.8.

A	B	C	S
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Tabela 3.8

Expressão extraída da tabela da verdade:

$$S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + A\overline{B}\,\overline{C} + AB\overline{C}$$

Transpondo a tabela para o diagrama, temos:

	$\overline{B}$		B	
	$\overline{C}$	C	C	$\overline{C}$
$\overline{A}$	Caso 0 1	Caso 1 0	Caso 3 1	Caso 2 1
A	Caso 4 1	Caso 5 0	Caso 7 0	Caso 6 1

Figura 3.21

Para efetuarmos a simplificação, seguimos o mesmo processo visto anteriormente, somente que, para 3 variáveis, os agrupamentos possíveis são os seguintes:

a) Oitava:

Agrupamento máximo, onde todas as localidades valem 1. A figura 3.22 apresenta esta situação:

Figura 3.22

b) Quadras:

Quadras são agrupamentos de 4 regiões, onde S é igual a 1, adjancentes ou em seqüência. Vamos agora formar algumas quadras possíveis num diagrama de 3 variáveis, a título de exemplo:

Figura 3.23 - (a) Quadra $\overline{A}$.
(b) Quadra $\overline{B}$.
(c) Quadra $\overline{C}$.

c) Pares:

A figura 3.24 apresenta, como exemplo, 2 pares entre os 12 possíveis em um diagrama de 3 variáveis:

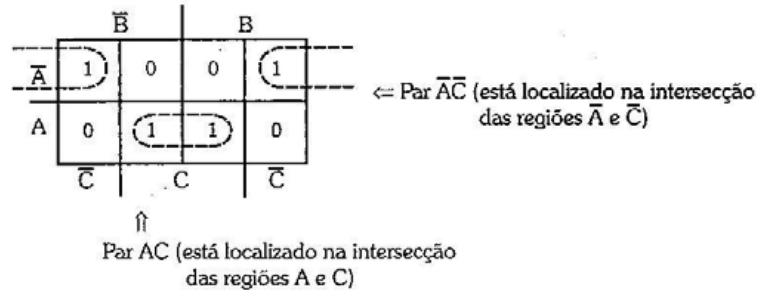



Figura 3.24

d) Termos isolados:

Vejamos na figura 3.25, alguns exemplos de termos isolados, que, como já dissemos, são os casos de entrada sem simplificação.

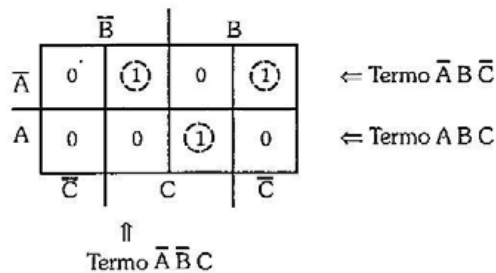



Figura 3.25

Para o exemplo, agrupamos primeiramente uma quadra e, logo após, um par, conforme mostra a figura 3.26.

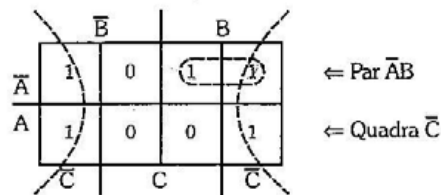



Figura 3.26

Notamos que esse par não depende de C, pois está localizado tanto em C como em $\overline{C}$, resultando sua expressão independente de C, ou seja, o termo $\overline{A}B$.

O passo final é somarmos as expressões referentes aos agrupamentos. A expressão final minimizada será: $S = \overline{A}B + \overline{C}$.

Como outro exemplo, vamos minimizar o circuito que executa a tabela 3.9.

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

Tabela 3.9

Transpondo para o diagrama, temos:

	$\overline{B}$	$\overline{B}$	B	B
$\overline{A}$	0	1	1	0
A	1	1	0	1
	$\overline{C}$	C	C	$\overline{C}$

Figura 3.27

Efetuando os agrupamentos, notamos que obtemos apenas 3 pares:

Figura 3.28

A expressão minimizada será: $S = \overline{A}C + A\overline{B} + A\overline{C}$.

Poderíamos também ter agrupado de outra maneira, conforme mostra a figura 3.29.

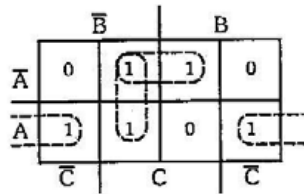



Figura 3.29

A expressão gerada, seria, então: $S = \overline{A}C + A\overline{C} + \overline{B}C$.

Estas duas expressões, aparentemente diferentes, possuem o mesmo comportamento em cada possibilidade, fato este comprovado, levantando-se as respectivas tabelas da verdade.

3.9.3 Diagrama de Veitch-Karnaugh para 4 Variáveis

O diagrama para 4 variáveis é visto na figura 3.30.

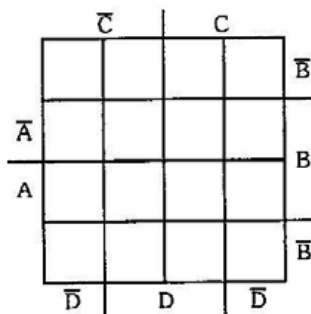



Figura 3.30

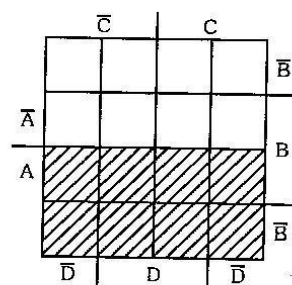
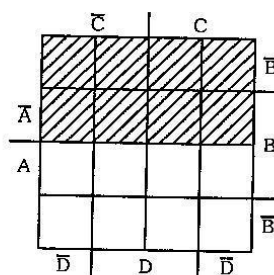
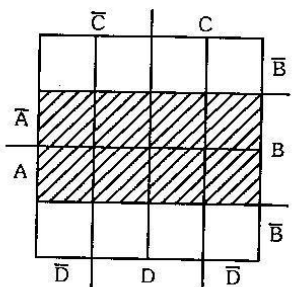
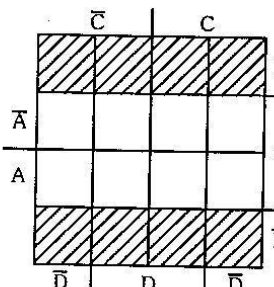
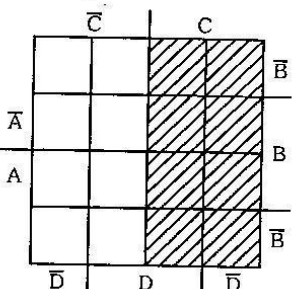
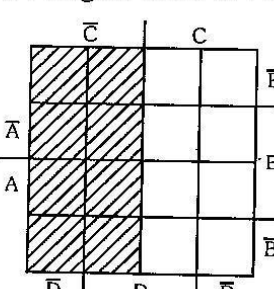
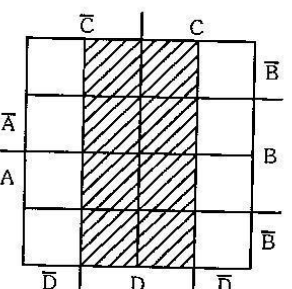
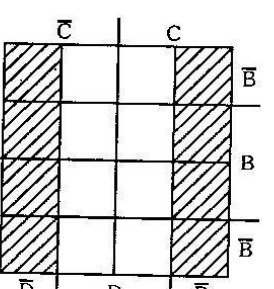
A figura 3.31 mostra as regiões assumidas pelas variáveis A, B, C e D neste mapa.

(a) Região onde A = 1.

(b) Região onde $\overline{A}$ = 1 (A = 0).

(c) Região onde B = 1.

(d) Região onde $\overline{B}$ = 1 (B = 0).

(e) Região onde C = 1.

(f) Região onde $\overline{C}$ = 1 (C = 0).

(g) Região onde D = 1.

(h) Região onde $\overline{D}$ = 1 (D = 0).

Figura 3.31

Neste tipo de diagrama, também temos uma região para cada caso da tabela da verdade, como podemos verificar no diagrama completo, visto na figura 3.32.

Casos	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Tabela 3.10

	$\overline{C}$		C		
$\overline{A}$	Caso 0 0 0 0 0 $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$	Caso 1 0 0 0 1 $\overline{A}\,\overline{B}\,\overline{C}\,D$	Caso 3 0 0 1 1 $\overline{A}\,\overline{B}\,C\,D$	Caso 2 0 0 1 0 $\overline{A}\,\overline{B}\,C\,\overline{D}$	$\overline{B}$
$\overline{A}$	Caso 4 0 1 0 0 $\overline{A}\,B\,\overline{C}\,\overline{D}$	Caso 5 0 1 0 1 $\overline{A}\,B\,\overline{C}\,D$	Caso 7 0 1 1 1 $\overline{A}\,B\,C\,D$	Caso 6 0 1 1 0 $\overline{A}\,B\,C\,\overline{D}$	B
A	Caso 12 1 1 0 0 $A\,B\,\overline{C}\,\overline{D}$	Caso 13 1 1 0 1 $A\,B\,\overline{C}\,D$	Caso 15 1 1 1 1 $A\,B\,C\,D$	Caso 14 1 1 1 0 $A\,B\,C\,\overline{D}$	B
A	Caso 8 1 0 0 0 $A\,\overline{B}\,\overline{C}\,\overline{D}$	Caso 9 1 0 0 1 $A\,\overline{B}\,\overline{C}\,D$	Caso 11 1 0 1 1 $A\,\overline{B}\,C\,D$	Caso 10 1 0 1 0 $A\,\overline{B}\,C\,\overline{D}$	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

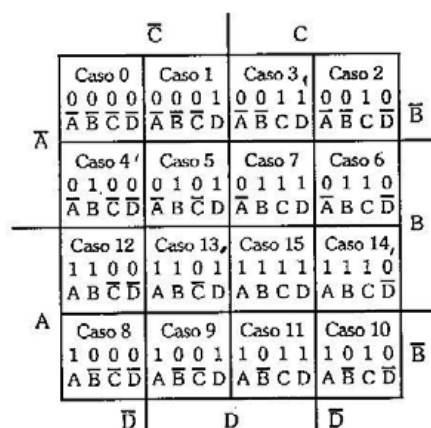



Figura 3.32

Vamos analisar a colocação de uma das possibilidades, vito que as outras são análogas.

Tomemos, como exemplo, o caso 8:

$$A\overline{B}\,\overline{C}\,\overline{D} \rightarrow 1000$$

$$A = 1,\ B = 0\ (\overline{B} = 1),\ C = 0\ (\overline{C} = 1)\ \text{e}\ D = 0\ (\overline{D} = 1)$$

Da intersecção dessas regiões, obtemos a região $A\overline{B}\,\overline{C}\,\overline{D}$, que é a referente ao caso 8:

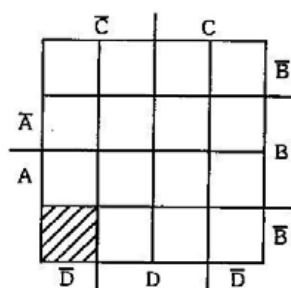



Figura 3.33

Para esclarecermos melhor a colocação do diagrama e analisarmos outros casos, vamos transpor para o mesmo a tabela 3.11.

A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Tabela 3.11

Expressão de S, extraída da tabela da verdade:

$$S = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BCD + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D$$
$$+ A\overline{B}CD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABCD$$

Transpondo a tabela para o diagrama, temos:

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	0	1	1	1	$\overline{B}$
$\overline{A}$	0	1	1	0	B
A	1	1	1	0	B
A	1	1	1	0	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

Figura 3.34

Para efetuarmos a simplificação, seguimos o mesmo processo para os diagramas de 3 variáveis, somente que neste caso, o principal agrupamento será a oitava.

Devemos ressaltar aqui, que no diagrama, os lados extremos opostos se comunicam, ou seja, podemos formar oitavas, quadras e pares com os termos localizados nos lados extremos opostos.

Vamos, como exemplo, verificar alguns desses casos no diagrama:

a) Exemplos de Pares:

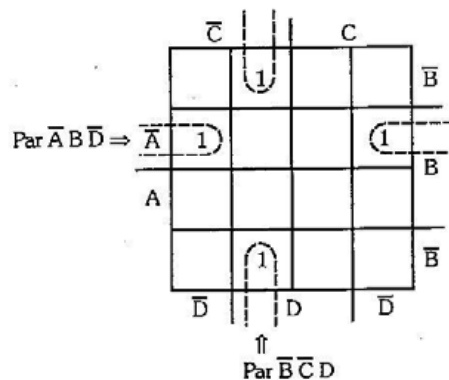



Figura 3.35

b) Exemplos de Quadras:

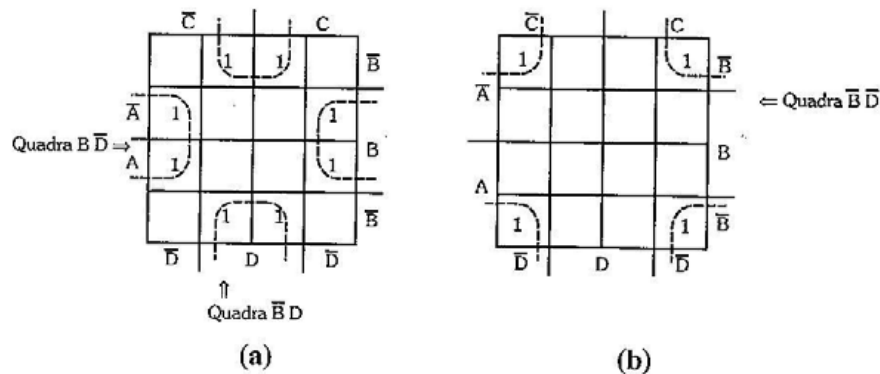



Figura 3.36

c) Exemplos de Oitavas:

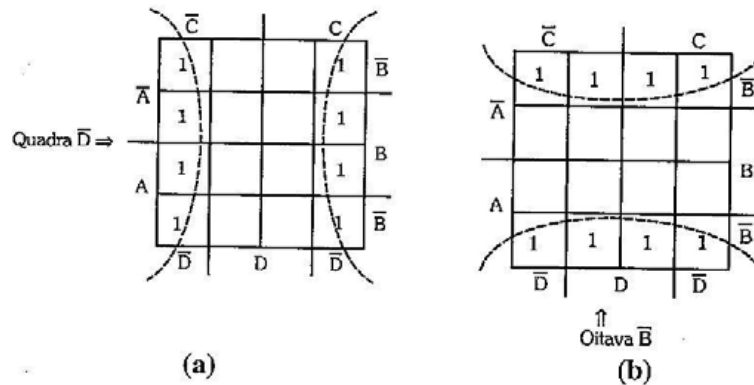



Figura 3.37

Convém observar que, neste mapa, as oitavas representam as próprias regiões A, $\overline{A}$, B, $\overline{B}$, C, $\overline{C}$, D e $\overline{D}$ e que o agrupamento máximo (mapa totalmente preenchido com 1) constitui-se em uma hexa, ou seja, agrupamento com 16 regiões valendo 1.

Após essa ressalva, vamos minimizar a expressão do nosso exemplo. Inicialmente, agrupamos as oitavas, em seguida as quadras, a seguir os pares e, por último, os termos isolados, se existirem. Expressões dos agrupamentos:

oitava: D

quadra : $A\overline{C}$

par: $\overline{A}\,\overline{B}\,C$

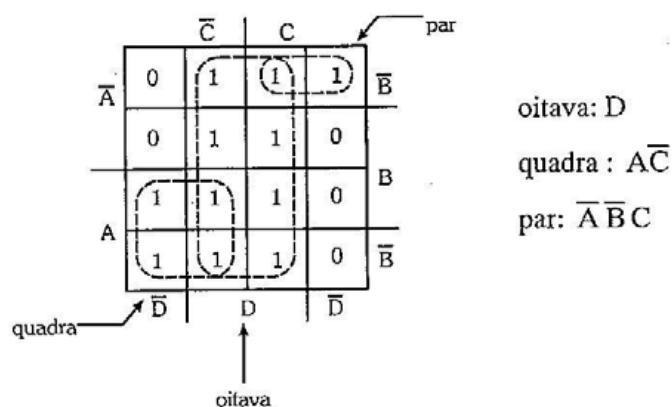



Figura 3.38

Somando as expressões, teremos a expressão final minimizada: $S = D + A\overline{C} + \overline{A}\,\overline{B}\,C$.

Como outro exemplo, vamos minimizar o circuito que executa a tabela 3.12.

A	B	C	D	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Tabela 3.12

Transpondo a tabela da verdade para o diagrama, temos:

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	0	1	1	0	$\overline{B}$
$\overline{A}$	1	1	1	1	B
A	0	0	1	0	B
A	0	0	0	1	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

Figura 3.39

No diagrama, temos: 2 quadras, 1 par e 1 termo isolado.

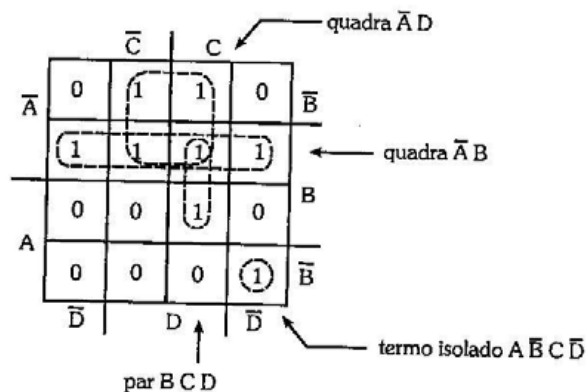



Figura 3.40

A expressão minimizada de S será a soma de todos esses agrupamentos:

$$S = A\overline{B}C\overline{D} + BCD + \overline{A}B + \overline{A}D$$

3.9.4 Exercícios Resolvidos

1 - Simplifique as expressões obtidas das tabelas a seguir, utilizando os diagramas de Veitch-Karnaugh.

a)

A	B	C	S
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Tabela 3.13

Transpondo para o diagrama de 3 variáveis e reconhecendo os agrupamentos, temos:

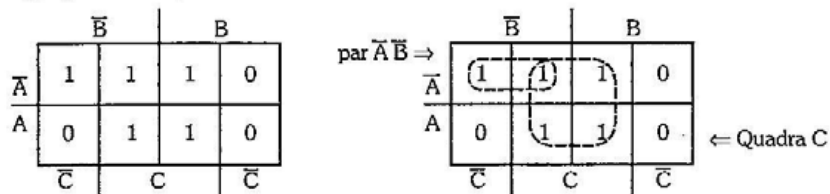



Figura 3.41

A expressão minimizada será: $S = C + \overline{A}\,\overline{B}$

b)

A	B	C	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Tabela 3.14

Transpondo para o diagrama e agrupando, temos:

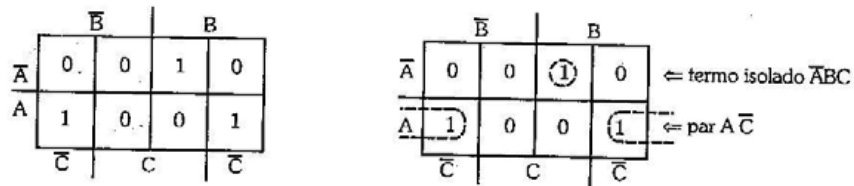



Figura 3.42

$\therefore S = A\overline{C} + \overline{A}BC$

c)

A	B	C	D	S
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	1
1	1	1	1	1

Tabela 3.15

Transpondo da tabela para o diagrama, temos:

	$\overline{C}$		C		
	1	0	0	1	$\overline{B}$
$\overline{A}$	1	1	1	1	B
A	1	0	1	1	B
	1	0	0	1	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

Figura 3.43

Agrupando o diagrama, temos:

Figura 3.44

A expressão minimizada será: $S = \overline{A}B + BC + \overline{D}$.

d)

A	B	C	D	S
0	0	0	0	1
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0

Tabela 3.16 (parte)

A	B	C	D	S
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Tabela 3.16

Transpondo para o diagrama, temos:

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$	
$\overline{A}$	1	1	1	0	$\overline{B}$
$\overline{A}$	1	0	0	1	B
A	0	0	1	0	B
A	0	0	1	1	$\overline{B}$

Figura 3.45

Podemos agrupar da seguinte maneira:

Figura 3.46

Neste diagrama, temos 5 pares gerando a expressão:

$$S = \overline{A}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}D + \overline{A}B\overline{D} + ACD + A\overline{B}C$$

Também podemos agrupar desta outra maneira:

Figura 3.47

Da mesma forma, gerando a expressão:

$$S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{D} + \overline{B}CD + ACD + A\overline{B}C$$

Podemos notar que simplificamos a expressão S por dois modos de agrupamentos, obtendo dois resultados aparentemente diferentes. Se analisarmos esses resultados nas respectivas tabelas da verdade, veremos que terão o mesmo comportamento.

Expressão simplificada de S:

$$S = \overline{A}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}D + \overline{A}B\overline{D} + ACD + A\overline{B}C$$

ou

$$S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{D} + \overline{B}CD + ACD + A\overline{B}C$$

2 - Minimize as expressões a seguir, utilizando os diagramas de Veitch-Karnaugh:

a) $S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}B\overline{C} + \overline{A}BC + ABC$

$AB + \overline{A}\,\overline{B}$

Colocando os termos diretamente no diagrama, temos:

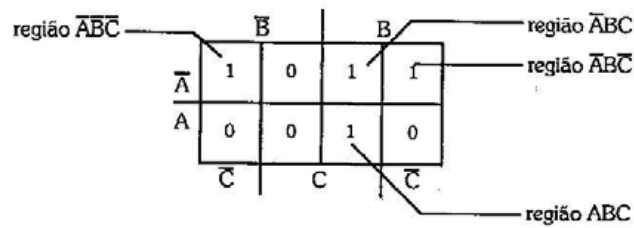



Figura 3.48

Temos, neste diagrama, 2 pares:

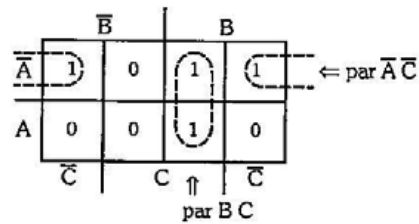



Figura 3.49

A expressão minimizada será: $S = \overline{A}\,\overline{C} + BC$

b) $S = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}\,\overline{B}C\overline{D} + \overline{A}B\overline{C}D + A\overline{B}\,\overline{C}\,\overline{D}$
$+ A\overline{B}\,\overline{C}D + A\overline{B}C\overline{D} + AB\overline{C}D + ABCD$

Transportando diretamente a expressão para o diagrama, temos:

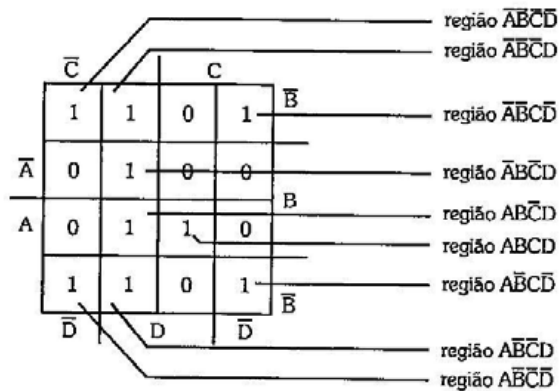



Figura 3.50

Agrupando os termos no diagrama, temos:

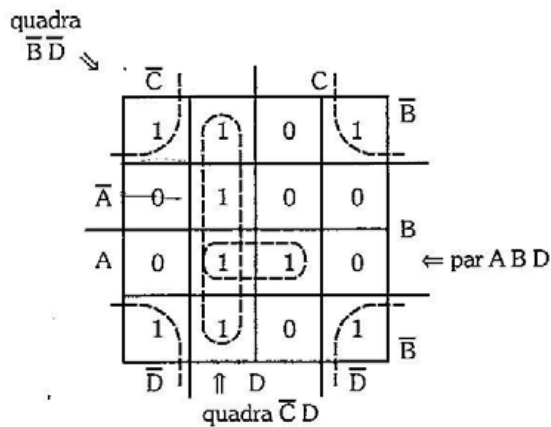



Figura 3.51

A expressão simplificada será: $S = ABD + \overline{C}D + \overline{B}\,\overline{D}$

c) $S = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}C\overline{D}$

$+ A\overline{B}CD + AB\overline{C}\,\overline{D} + AB\overline{C}D + ABC\overline{D} + ABCD$

Passando a expressão para o diagrama, temos:

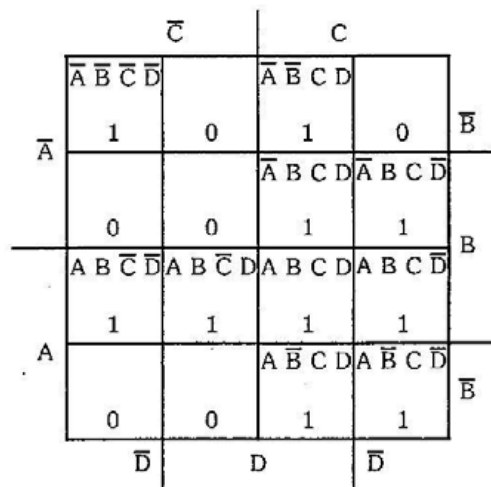



Figura 3.52

Efetuando os agrupamentos, temos:

1 termo isolado: $\overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$

4 quadras: AB, CD, BC e AC

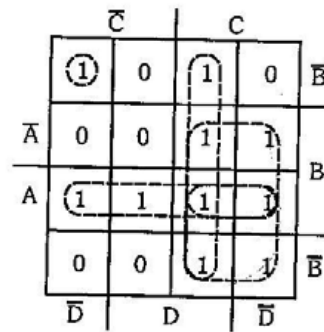



Figura 3.53

É importante ressaltar que uma oitava agrupada representa maior simplificação que uma quadra, e uma quadra agrupada maior simplificação que um par, e este, maior simplificação que um termo isolado. Assim sendo, deve-se preferir agrupar em oitavas, e se não for possível, em quadras e também se não for possível, em pares, mesmo que alguns casos já tenham sido considerados em outros agrupamentos, lembrando sempre, que devemos ter o menor número de agrupamentos possível.

A expressão final minimizada será:

$$S = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + AB + CD + BC + AC$$

3.9.5 Diagrama para 5 Variáveis

O diagrama de Veitch-Karnaugh para simplificar expressões com 5 variáveis de entrada é visto na figura 3.54.

Ā | A

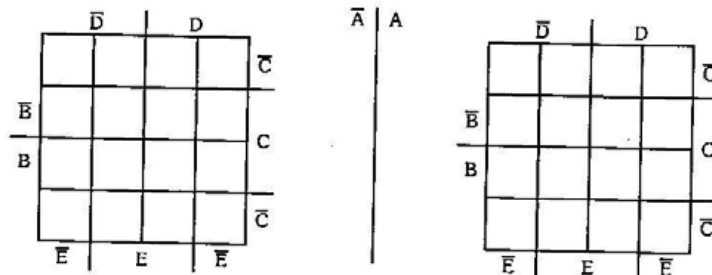



Figura 3.54

Vamos verificar algumas das regiões deste diagrama:

a) Região onde A = 1:

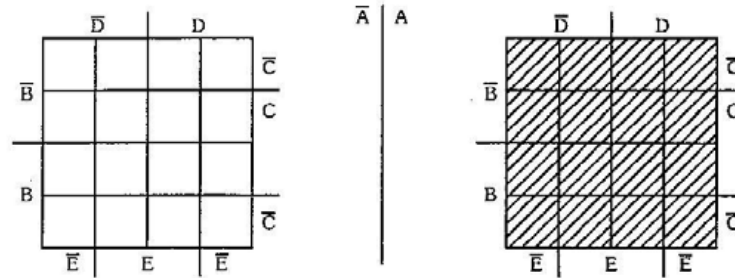



Figura 3.55

b) Região onde B = 1:

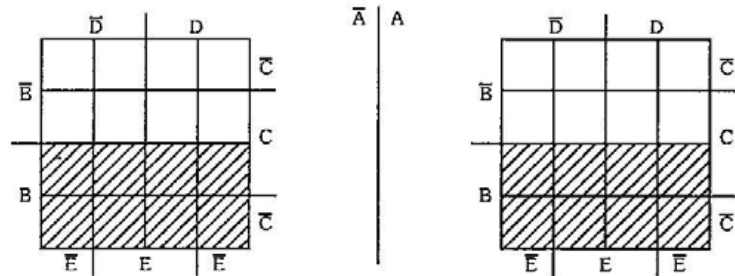



Figura 3.56

c) Região onde C = 1:

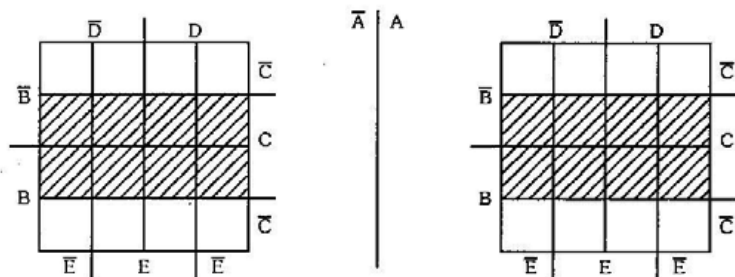



Figura 3.57

d) Região onde D = 1:

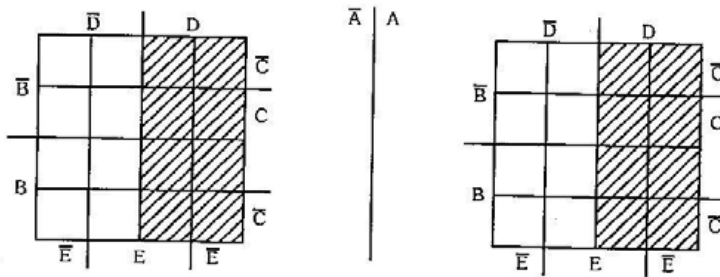



Figura 3.58

e) Região onde E = 1:

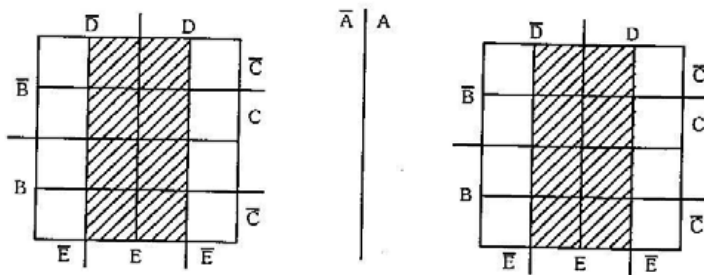



Figura 3.59

De forma análoga, o diagrama possui as regiões relativas às variáveis opostas às mostradas, ou seja, $\overline{A}$, $\overline{B}$, $\overline{C}$, $\overline{D}$ e $\overline{E}$. Todas estas regiões denominam-se hexas.

A colocação de uma condição, neste diagrama, faz-se de maneira análoga às anteriores.

Vamos, por exemplo, verificar a região onde: A = 1, B = 0, C = 1, D = 0 e E = 0, ou seja, $A\,\overline{B}\,C\,\overline{D}\,\overline{E}$:

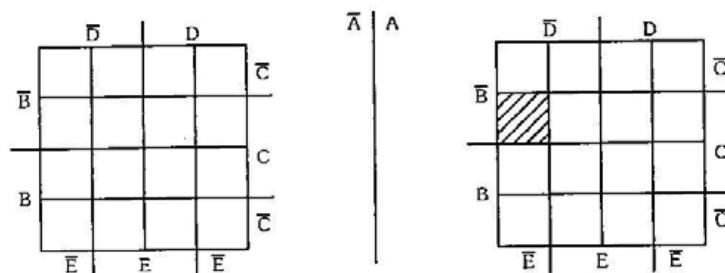



Figura 3.60

Para efetuarmos a simplificação num diagrama de 5 variáveis, devemos tentar primeiramente em hexas, em seguida em oitavas, em quadras, em pares e por último em termos isolados.

Para visualizarmos melhor as hexas, oitavas, quadras e pares, devemos enxergar o diagrama da esquerda sobreposto ao da direita, conforme mostra a figura 3.61.

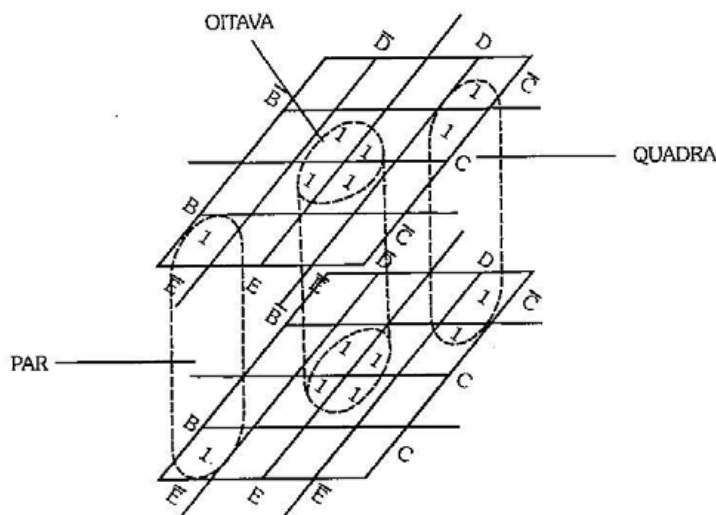



Figura 3.61

Podemos visualizar, por exemplo, que o par, a oitava e a quadra formam-se nos dois planos.

Vamos, agora, fazer a transposição e a simplificação da tabela 3.17, para melhor entendimento destes conceitos.

A	B	C	D	E	S
0	0	0	0	0	1
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	1
0	1	1	1	0	1
0	1	1	1	1	0
1	0	0	0	0	0
1	0	0	0	1	0
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	0	1	1
1	0	1	1	0	1
1	0	1	1	1	0
1	1	0	0	0	0
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	0
1	1	1	0	0	1
1	1	1	0	1	1
1	1	1	1	0	1
1	1	1	1	1	1

Tabela 3.17

Transpondo para o diagrama, temos:

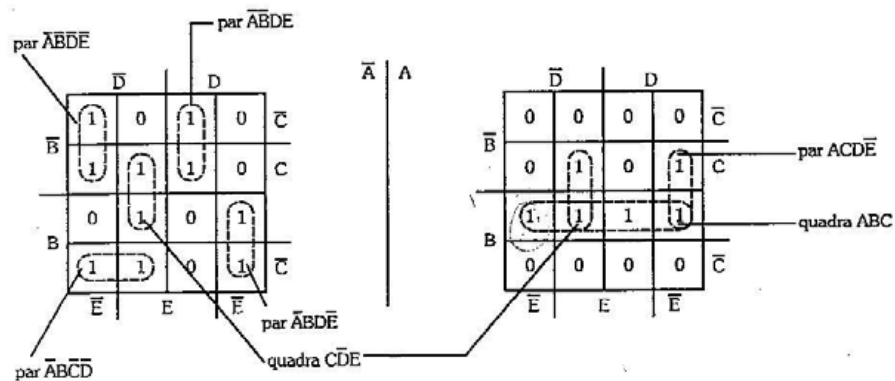



Figura 3.62

Resumindo os agrupamentos obtidos, temos:

$$2 \text{ quadras: } \begin{cases} C\overline{D}E \\ ABC \end{cases}$$

$$5 \text{ pares: } \begin{cases} \overline{A}\,\overline{B}\,\overline{D}\,\overline{E} \\ \overline{A}B\overline{C}\,\overline{D} \\ \overline{A}BD\overline{E} \\ \overline{A}\,\overline{B}DE \\ ACD\overline{E} \end{cases}$$

A expressão minimizada será:

$$S = C\overline{D}E + ABC + \overline{A}\,\overline{B}\,\overline{D}\,\overline{E} + \overline{A}B\overline{C}\,\overline{D} + \overline{A}BD\overline{E} + \overline{A}\,\overline{B}DE + ACD\overline{E}$$

3.9.5.1 Exercício Resolvido

Simplifique a expressão da tabela 3.18.

A	B	C	D	E	S
0	0	0	0	0	1
0	0	0	0	1	1
0	0	0	1	0	0
0	0	0	1	1	0
0	0	1	0	0	1
0	0	1	0	1	1
0	0	1	1	0	0
0	0	1	1	1	0
0	1	0	0	0	1
0	1	0	0	1	0
0	1	0	1	0	0
0	1	0	1	1	0
0	1	1	0	0	0
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	0
1	0	0	0	0	1
1	0	0	0	1	1
1	0	0	1	0	0
1	0	0	1	1	0
1	0	1	0	0	1
1	0	1	0	1	1
1	0	1	1	0	1
1	0	1	1	1	0
1	1	0	0	0	1
1	1	0	0	1	0
1	1	0	1	0	1
1	1	0	1	1	1
1	1	1	0	0	0
1	1	1	0	1	0
1	1	1	1	0	1
1	1	1	1	1	1

Tabela 3.18

Numerando os casos das 32 possibilidades das 5 variáveis de 0 a 31, obtemos a localização no diagrama, vista na figura 3.63.

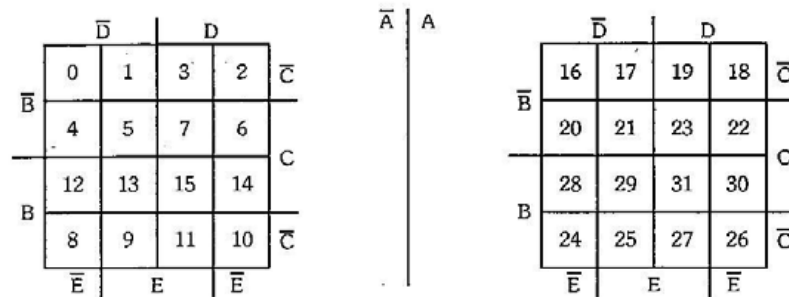



Figura 3.63

Colocando os casos no diagrama, temos:

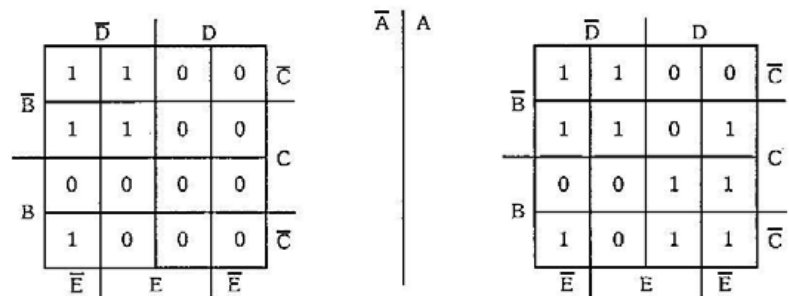



Figura 3.64

Os agrupamentos para obtenção da expressão final simplificada são vistos na figura 3.65.

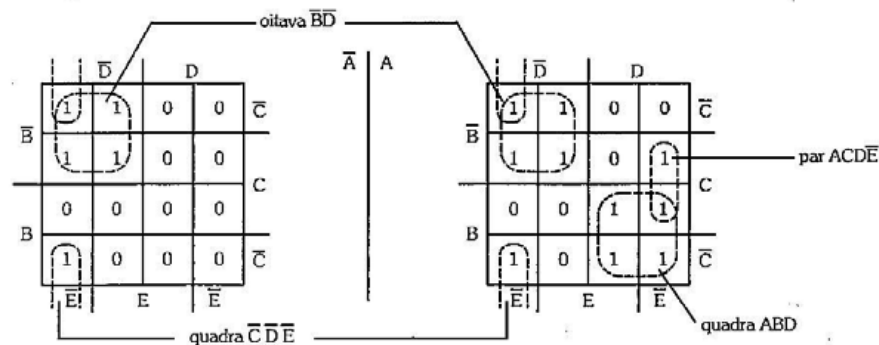



Figura 3.65

A expressão simplificada será: $S = \overline{B}\,\overline{D} + \overline{C}\,\overline{D}\,\overline{E} + ABD + ACD\overline{E}$.

3.9.6 Diagramas com Condições Irrelevantes

Chamamos de condição irrelevante (X) a situação de entrada onde a saída pode assumir 0 ou 1 indiferentemente. Esta condição ocorre principalmente pela impossibilidade prática do caso de entrada acontecer, sendo utilizada em várias situações nos capítulos posteriores. Para sua utilização em diagramas de Veitch-Karnaugh, devemos, para cada condição irrelevante, adotar 0 ou 1, dos dois, aquele que possibilitar melhor agrupamento e conseqüentemente maior simplificação.

Para esclarecer este processo, vamos utilizar a tabela 3.19.

A	B	C	S
0	0	0	X
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

Tabela 3.19

Transpondo esta tabela para o diagrama, temos:

	$\overline{B}$		B	
$\overline{A}$	X	1	1	1
A	0	0	0	0
	$\overline{C}$	C		$\overline{C}$

Figura 3.66

O símbolo (X) indica que neste caso a saída pode assumir 0 ou 1, indiferentemente, pois, ou a situação de entrada é impossível de acontecer, ou, ainda, possibilita qualquer dos 2 valores na saída. Para fins de simplificação, devemos adotar X = 1, pois assim sendo, agrupamos uma quadra, ao invés de 2 pares (no caso de X = 0), representando uma maior simplificação da expressão de saída: $S = \overline{A}$.

Convém ressaltar que, em uma tabela da verdade, podemos ter várias condições irrelevantes que devem ser consideradas independentemente, conforme agrupamento em que se encontram. Para exemplificar, vamos simplificar a expressão extraída da tabela 3.20.

A	B	C	D	S
0	0	0	0	X
0	0	0	1	0
0	0	1	0	1
0	0	1	1	X
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	X
1	0	1	1	0
1	1	0	0	0
1	1	0	1	X
1	1	1	0	0
1	1	1	1	X

Tabela 3.20

Passando para o diagrama de 4 variáveis, temos:

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$	
$\overline{A}$	X	0	X	1	$\overline{B}$
$\overline{A}$	1	0	1	1	B
A	0	X	X	0	B
A	0	1	0	X	$\overline{B}$

Figura 3.67

O próximo passo é agrupar as regiões que valem 1, utilizando a condição irrelevante (X) para completar o agrupamento. Convém lembrar que, para maior simplificação, devemos ter um número mínimo de agrupamentos, cada um deles, porém, com o maior número de células possível. Assim sendo, temos:

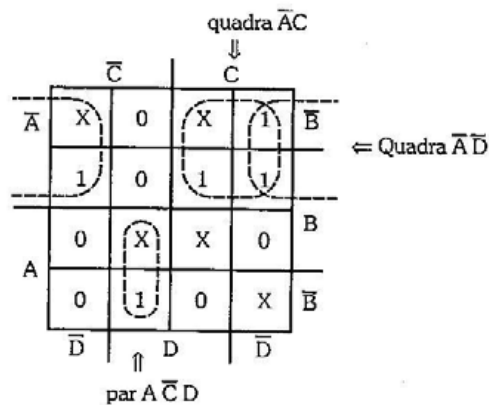



Figura 3.68

Notamos, na figura 3.68, que as condições irrelevantes pertencentes aos agrupamentos receberam valor 1, enquanto as deixadas de fora, valor 0.

A expressão simplificada será composta por 2 quadras e um par, sendo esta:

$$S = \overline{A}C + \overline{A}\,\overline{D} + A\overline{C}D.$$

3.9.6.1 Exercícios Resolvidos

1 - A tabela 3.21 representa as possibilidades de saída obtidas de um projeto envolvendo 3 variáveis A, B e C. Determine a expressão simplificada.

A	B	C	S
0	0	0	1
0	0	1	X
0	1	0	0
0	1	1	1
1	0	0	X
1	0	1	1
1	1	0	X
1	1	1	X

Tabela 3.21

A figura 3.69 apresenta a colocação dos valores da tabela no diagrama (a) e os respectivos agrupamentos para a obtenção da expressão simplificada (b).

(a)

(b)

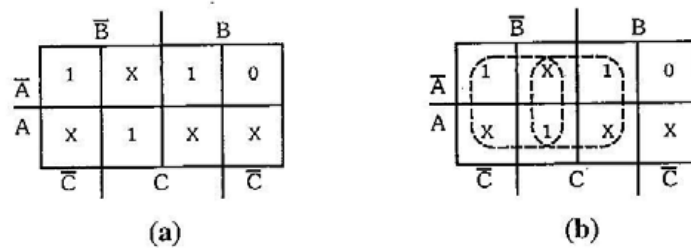



Figura 3.69

A expressão simplificada será composta pelas 2 quadras obtidas: $S = \overline{B} + C$.

2 - Simplifique a expressão representativa da tabela 3.22.

A	B	C	D	S
0	0	0	0	1
0	0	0	1	X
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	0
1	0	1	0	X
1	0	1	1	1
1	1	0	0	X
1	1	0	1	1
1	1	1	0	X
1	1	1	1	0

Tabela 3.22

Passando os valores da tabela para o diagrama, temos:

	$\overline{C}$	$\overline{C}$	C	C	
$\overline{A}$	1	X	0	1	$\overline{B}$
$\overline{A}$	1	1	0	1	B
A	X	1	0	X	B
A	1	0	1	X	$\overline{B}$
	$\overline{D}$	D	D	$\overline{D}$	

Figura 3.70

Logo após, efetuamos os agrupamentos visando obter a expressão simplificada de forma máxima:

Figura 3.71

Como resultado, obtemos a oitava $\overline{D}$, a quadra $B\overline{C}$ e o par $A\overline{B}C$, gerando a expressão: $S = \overline{D} + B\overline{C} + A\overline{B}C$.

É importante observar que se tivéssemos agrupado precipitadamente, ao início do exercício a quadra $\overline{A}\,\overline{C}$, geraríamos erradamente um termo a mais na expressão final. Para melhor condução do processo de agrupamento, devemos iniciar sempre pelos agrupamentos obrigatórios e bem-definidos.

3.9.7 Casos que não Admitem Simplificação

Vamos, neste tópico, efetuar uma análise das expressões representativas das funções OU Exclusivo e Coincidência, no que se refere às suas colocações nos diagramas de Veitch-Karnaugh.

A figura 3.72 mostra a colocação destas expressões nos diagramas, no caso de 2 variáveis.

Figura 3.72

(a) $S = A \oplus B = \overline{A}B + A\overline{B}$

(b) $S = A \odot B = \overline{A}\,\overline{B} + AB$

Pela figura, notamos que as expressões encontram-se na forma de máxima simplificação, não havendo outra possibilidade, pois em cada diagrama temos 2 termos isolados que são as próprias expressões de entrada.

No caso de utilizarmos 3 variáveis, as expressões são, respectivamente, $S = A \oplus B \oplus C$ e $S = A \odot B \odot C$. Para levantarmos suas tabelas da verdade, devemos tomar as variáveis de 2 em 2, ou seja, efetuar primeiro as operações entre 2 das variáveis e com o resultado obtido efetuar a operação com a terceira variável. Esse processo se deve ao fato de as funções OU Exclusivo e Coincidência não serem válidas para mais de 2 variáveis de entrada, podendo ser aplicado, tomando primeiramente 2 quaisquer das 3 variáveis da expressão, indiferentemente. As tabelas 3.23 e 3.24 mostram os resultados das operações $S = A \oplus B \oplus C$ e $S = A \odot B \odot C$, em todas as possibilidades.

A	B	C	$(A \oplus B) \oplus C$	$A \oplus (B \oplus C)$	$(A \oplus C) \oplus B$
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	1	1	1

Tabela 3.23

A	B	C	$(A \odot B) \odot C$	$A \odot (B \odot C)$	$(A \odot C) \odot B$
0	0	0	0	0	0
0	0	1	1	1	1
0	1	0	1	1	1
0	1	1	0	0	0
1	0	0	1	1	1
1	0	1	0	0	0
1	1	0	0	0	0
1	1	1	1	1	1

Tabela 3.24

Passando a coluna S (iguais em todos os casos) para o diagrama, temos

Figura 3.73

Da mesma forma, temos apenas termos isolados, não havendo possibilidade de simplificação.

Extraindo a expressão da tabela inicial ou do diagrama, temos:

$$S = \overline{A}\,\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC.$$

Evidenciando $\overline{A}$ e A, temos:

$$S = \overline{A}(\overline{B}C + B\overline{C}) + A(\overline{B}\,\overline{C} + BC)$$

Substituindo-se os parênteses respectivamente por: $B \oplus C$ e $B \odot C$, temos:

$$S = \overline{A}\,(B \oplus C) + A\,(B \odot C)$$

Como $B \odot C = \overline{B \oplus C}$, reescrevemos:

$$S = \overline{A}(B \oplus C) + A\,(\overline{B \oplus C})$$

Chamando $(B \oplus C)$ de X, temos:

$$S = \overline{A}X + A\overline{X} = A \oplus X$$

Substituindo X, temos:

$$S = A \oplus B \oplus C$$

Inicialmente, se tivéssemos evidenciado outras variáveis, teríamos outras ordens no resultado, de conformidade com as tabelas levantadas. Ainda, se tivéssemos substituído $B \oplus C$ por $\overline{(B \odot C)}$, obteríamos $S = A \odot B \odot C$, que analogamente, conforme as tabelas é equivalente a $S = A \oplus B \oplus C$.

Se estendermos o estudo para mais variáveis, obteremos:

Para 4 variáveis: $S = \overline{A \oplus B \oplus C \oplus D} \quad = \quad A \odot B \odot C \odot D$

Para 5 variáveis: $S = A \oplus B \oplus C \oplus D \oplus E \quad = \quad A \odot B \odot C \odot D \odot E$

De posse de resultados, concluímos que para um número de par de variáveis, temos a função OU Exclusivo como sendo o complemento da função Coincidência e para um número ímpar de variáveis temos a função OU Exclusivo como sendo igual à função Coincidência.

3.9.8 Agrupamentos de Zeros

Podemos, alternativamente, agrupar as células que valem 0 para obtermos a expressão simplificada em diagramas de Veitch-Karnaugh, porém, com esta prática, obtemos o complemento da função, ou seja, a saída $\overline{S}$. Para ilustrar esta situação, vamos simplificar a expressão da tabela 3.25.

A	B	C	S
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

Tabela 3.25

Passando para o diagrama e efetuando o agrupamento, temos:

	$\overline{B}$		B	
$\overline{A}$	0	1	1	0
A	1	1	1	1
	$\overline{C}$	C	C	$\overline{C}$

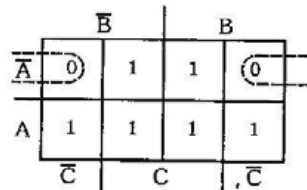



Figura 3.74

Pela figura notamos que obtemos um par formado por zeros. Conforme o exposto, a expressão será: $\overline{S} = \overline{A}\,\overline{C}$, sendo $S = \overline{(\overline{A}\,\overline{C})}$.

Aplicando o teorema De Morgan a esta expressão, temos: $S = \overline{(\overline{A}.\overline{C})} = A + C$.

$\therefore S = A + C$

Convém observar que a mesma expressão seria obtida, resultado dos agrupamentos de 2 quadras, se houvéssemos utilizado o procedimento convencional anteriormente visto.

3.9.9 Outra Forma de Apresentação do Diagrama de Veitch-Karnaugh

Ao invés de representarmos o diagrama dividindo-o em regiões, como visto até aqui, podemos representá-lo de uma forma análoga, conforme a figura 3.75.

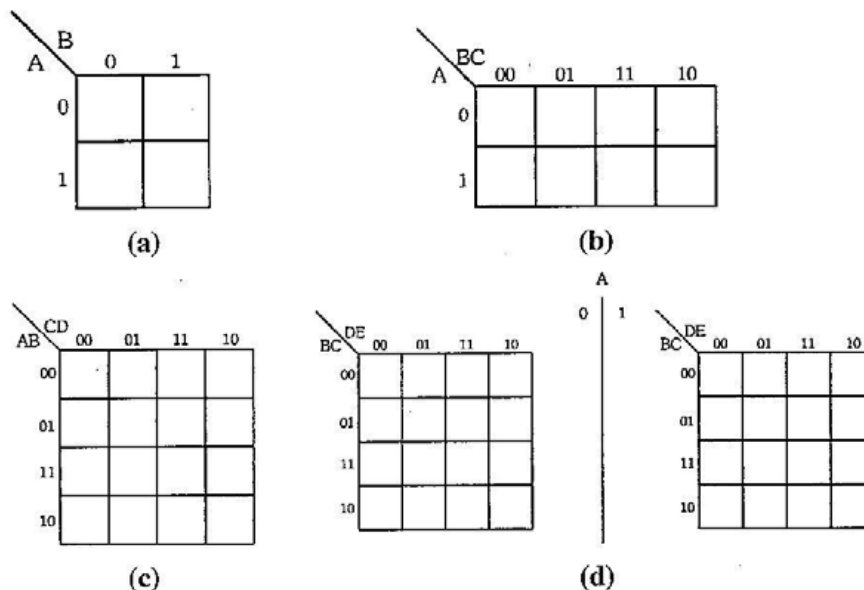



Figura 3.75 - (a) 2 variáveis.

(b) 3 variáveis.

(c) 4 variáveis.

(d) 5 variáveis.

Pela figura, podemos notar que os diagramas são semelhantes, possuindo apenas a identificação das regiões pelo valor assumido pela variável (exemplo: A=0 => região $\overline{A}$, A = 1 => região A). Tanto a colocação dos casos, bem como os agrupamentos obtidos se fazem de maneira análoga, levando aos mesmos resultados. A figura 3.76 apresenta os dois estilos dos diagramas de quatro variáveis sobrepostos, onde se observam claramente os níveis assumidos pelas variáveis, idênticos para ambos os mapas.

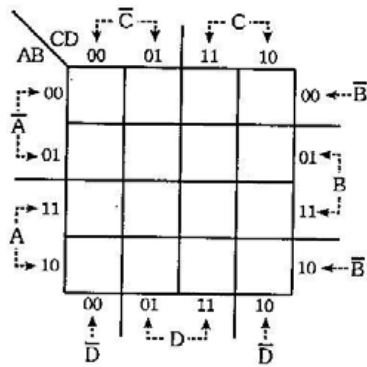



Figura 3.76

3.10 Exercícios Propostos

3.10.1 - Simplifique cada expressão, utilizando a Álgebra de Boole.

a) $S = AB\overline{C} + \overline{A}\,\overline{B}C + ABC + \overline{A}BC + \overline{A}B\overline{C}$

b) $S = AB\overline{C}D + \overline{A}\,\overline{B}C\overline{D} + AB\overline{C}\,\overline{D} + \overline{A}BC\overline{D} + ABC\overline{D} + A\overline{B}C\overline{D} + ABCD$

3.10.2 - Simplifique utilizando a Álgebra de Boole:

$$S = \overline{[\overline{(\overline{\overline{B}} + \overline{\overline{C}} + \overline{\overline{D}})}\,\overline{(\overline{\overline{A}} + B + \overline{C})} + C]} + \overline{A}\,\overline{B}C + \overline{B}\,\overline{(A+C)}$$

3.10.3 - Idem, para a expressão:

$$S = A[\overline{\overline{\overline{B}}\,\overline{(\overline{C} + D)} + \overline{A}\,\overline{\overline{(B + \overline{C})}}}] + C\overline{D} + A\overline{B}C + AB$$

3.10.4 - Idem, para a expressão:

$$S = \overline{(A \oplus B + \overline{B}C\overline{\overline{D}})}\,\overline{[\overline{D} + \overline{B}C + D\overline{(\overline{\overline{A}} + B)}]} + \overline{A}\,\overline{D}$$

3.10.5 - Idem, para a expressão:

$$S = \overline{[\overline{(B + C\overline{D} + \overline{\overline{D}} + AC)}\,(A + \overline{B} + \overline{C}) + \overline{B}\,\overline{\overline{(C + \overline{A}BC + AC)}}]}\,(A + B)$$

3.10.6 - Desenhe o circuito que.executa a expressão, simplificado.

$$S = \overline{(\overline{\overline{B}} + \overline{\overline{D}})}\,\overline{\{\overline{B} + C \odot D + \overline{A}\,\overline{\overline{[B\overline{C} + \overline{B}C + A + B\overline{(\overline{C} + \overline{\overline{D}})}]}}\}}$$

3.10.7 - Simplifique através da Álgebra de Boole:

$$S = (\overline{\bar{A}B + C\bar{D} + AD})\,\overline{\{\bar{B}\overline{[C \oplus D + \bar{A}(\overline{\bar{B} + \bar{C}}) + A\bar{B}\bar{C}]} + \bar{A}\}}$$

3.10.8 - Demonstre que:

$$A \odot (B \oplus C) = A \oplus (B \odot C)$$

3.10.9 - Através dos diagramas de Veitch-Karnaugh, determine a expressão simplificada de S_1 e S_2 da tabela 3.26.

A B	S_1	S_2
0 0	1	1
0 1	0	1
1 0	1	0
1 1	1	0

Tabela 3.26

3.10.10 - Simplifique as expressões de S_1, S_2, S_3 e S_4 da tabela 3.27, utilizando os mapas de Veitch-Karnaugh.

A	B	C	S_1	S_2	S_3	S_4
0	0	0	1	1	0	0
0	0	1	0	1	1	1
0	1	0	1	1	0	1
0	1	1	1	0	0	0
1	0	0	1	1	1	1
1	0	1	1	1	1	0
1	1	0	0	1	1	1
1	1	1	1	0	0	1

Tabela 3.27

3.10.11 - Idem ao anterior, para a tabela 3.28.

A	B	C	D	S_1	S_2	S_3	S_4
0	0	0	0	1	1	0	0
0	0	0	1	1	0	0	0
0	0	1	0	1	1	1	0
0	0	1	1	1	0	0	1
0	1	0	0	1	1	1	1
0	1	0	1	0	1	1	1
0	1	1	0	0	1	1	0
0	1	1	1	1	1	0	1
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	0	1	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	0	1	1	1
1	1	1	0	0	0	0	1
1	1	1	1	1	1	0	1

Tabela 3.28

3.10.12- Simplifique as expressões utilizando diagramas de Veitch-Karnaugh:

a) $S = A\overline{B}\,\overline{C} + A\overline{B}C + \overline{A}BC + \overline{A}B\overline{C} + ABC$

b) $S = \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + AB\overline{C}D + \overline{A}BCD + A\overline{B}\,\overline{C}D$
$ABCD + A\overline{B}\,\overline{C}\,\overline{D}$

c) $S = \overline{B}\,\overline{D} + \overline{A} + A\overline{B}\,\overline{C}D + A\overline{B}CD + \overline{A}\,\overline{C}$

d) $S = ABC + AB + \overline{A}BCD + BD + CD + \overline{B}C\overline{D} + \overline{A}B\overline{C}\,\overline{D}$

3.10.13 - Determine as expressões simplificadas para S_1 e S_2 da tabela 3.29.

A	B	C	D	E	S_1	S_2
0	0	0	0	0	1	1
0	0	0	0	1	1	0
0	0	0	1	0	1	1
0	0	0	1	1	1	0
0	0	1	0	0	0	1
0	0	1	0	1	1	1
0	0	1	1	0	0	1
0	0	1	1	1	1	1
0	1	0	0	0	0	1
0	1	0	0	1	0	0
0	1	0	1	0	1	1
0	1	0	1	1	0	0
0	1	1	0	0	1	1
0	1	1	0	1	1	1
0	1	1	1	0	0	1
0	1	1	1	1	1	1
1	0	0	0	0	1	1
1	0	0	0	1	1	0
1	0	0	1	0	0	1
1	0	0	1	1	0	1
1	0	1	0	0	0	1
1	0	1	0	1	1	1
1	0	1	1	0	0	1
1	0	1	1	1	1	1
1	1	0	0	0	0	1

Tabela 3.29 (parte)

A	B	C	D	E	S_1	S_2
1	1	0	0	1	0	0
1	1	0	1	0	1	1
1	1	0	1	1	0	0
1	1	1	0	0	0	1
1	1	1	0	1	1	1
1	1	1	1	0	0	1
1	1	1	1	1	1	1

Tabela 3.29

3.10.14 - Simplifique as expressões de S_1 e S_2 da tabela 3.30.

A	B	C	S_1	S_2
0	0	0	X	1
0	0	1	0	X
0	1	0	1	0
0	1	1	X	0
1	0	0	1	0
1	0	1	X	1
1	1	0	X	X
1	1	1	1	X

Tabela 3.30

3.10.15 - Determine as expressões simplificadas de S_1, S_2, S_3 e S_4 da tabela 3.31.

A	B	C	D	S_1	S_2	S_3	S_4
0	0	0	0	1	X	0	X
0	0	0	1	X	X	0	0
0	0	1	0	X	1	0	X
0	0	1	1	X	0	1	1
0	1	0	0	1	X	X	1
0	1	0	1	0	1	X	X
0	1	1	0	X	0	1	0
0	1	1	1	X	1	0	1
1	0	0	0	X	1	X	0
1	0	0	1	1	0	1	1
1	0	1	0	X	X	0	0
1	0	1	1	1	1	0	X
1	1	0	0	X	0	1	1
1	1	0	1	X	1	0	1
1	1	1	0	1	1	X	1
1	1	1	1	0	X	1	X

Tabela 3.31

3.10.16 - Desenhe os circuitos minimizados que executam as saídas S_1 e S_2 da tabela da verdade:

A	B	C	D	E	S_1	S_2
0	0	0	0	0	0	1
0	0	0	0	1	0	X
0	0	0	1	0	1	1
0	0	0	1	1	0	X
0	0	1	0	0	1	X
0	0	1	0	1	1	1
0	0	1	1	0	0	X
0	0	1	1	1	1	1
0	1	0	0	0	0	1
0	1	0	0	1	1	0
0	1	0	1	0	1	1
0	1	0	1	1	0	0
0	1	1	0	0	1	X
0	1	1	0	1	1	1
0	1	1	1	0	0	0
0	1	1	1	1	1	1
1	0	0	0	0	0	1
1	0	0	0	1	0	X
1	0	0	1	0	1	1
1	0	0	1	1	0	0
1	0	1	0	0	1	X
1	0	1	0	1	1	1
1	0	1	1	0	0	0
1	0	1	1	1	1	1
1	1	0	0	0	0	X
1	1	0	0	1	0	1
1	1	0	1	0	1	1
1	1	0	1	1	0	1
1	1	1	0	0	1	1
1	1	1	0	1	1	X
1	1	1	1	0	0	1
1	1	1	1	1	1	X

Tabela 3.32

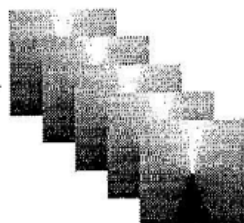
154

3.10.17 - Obtenha a expressão simplificada:

$$S = (\bar{A} + B)\overline{\{\bar{B} + (B \oplus C)[\overline{\bar{A}B\bar{C} + B\overline{(A + \bar{D})} + B\bar{C} + \bar{B}D}] + ABD\}}$$

3.10.18 - Prove que:

$$\overline{A \oplus B \oplus C \oplus D} = A \odot B \odot C \odot D$$

CAPÍTULO 4

Circuitos Combinacionais 1ª Parte

4.1 Introdução

Um dos capítulos importantes da Eletrônica Digital é o que trata dos circuitos combinacionais. É através do estudo destes que poderemos compreender o funcionamento de circuitos, tais como: somadores, subtratores, circuitos que executam prioridades, codificadores, decodificadores e outros muito utilizados na construção de computadores e em vários outros sistemas digitais.

O circuito combinacional é aquele em que a saída depende única e exclusivamente das combinações entre as variáveis de entrada.

Podemos utilizar um circuito lógico combinacional para solucionar problemas em que necessitamos de uma resposta, quando acontecerem determinadas situações, representadas pelas variáveis de entrada. Para construirmos estes circuitos, necessitamos de suas expressões características que como vimos no capítulo anterior, são obtidas das tabelas da verdade que representam as situações já mencionadas.

A figura 4.1 ilustra a seqüência do processo, onde, a partir da situação, obtemos a tabela da verdade e a partir desta, através das técnicas já conhecidas, a expressão simplificada e o circuito final.

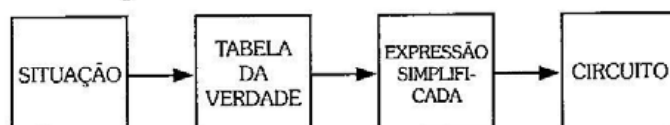



Figura 4.1

4.2 Projetos de Circuitos Combinacionais

Nos itens subseqüentes, mostraremos como obter um circuito para resolver um problema utilizando a Eletrônica Digital a partir de uma situação prática. Os projetos apresentados a seguir, embora simulem situações reais, são didáticos e servem para descrever o método de realização, podendo ser empregados na prática como modelos para a solução de pequenos problemas ou, ainda, para a construção de circuitos periféricos dentro de sistemas digitais mais complexos, utilizando circuitos integrados específicos e microprocessadores. A figura 4.2 mostra o esquema geral de um circuito combinacional composto pelas variáveis de entrada, o circuito propriamente dito e sua(s) saída(s).

Figura 4.2

Notamos que o circuito lógico pode possuir diversas variáveis de entrada e uma ou mais saídas conforme o caso do projeto. A seguir, estudaremos, a título de exemplo, casos de 2, 3 e 4 variáveis.

4.2.1 Circuitos com 2 Variáveis

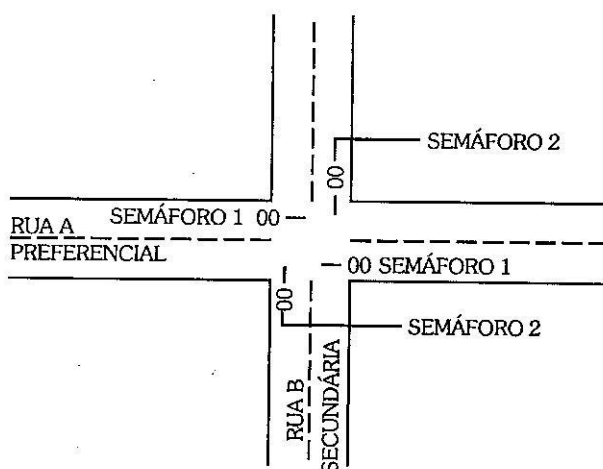



Figura 4.3

A figura 4.3 representa o cruzamento das ruas A e B. Neste cruzamento, queremos instalar um sistema automático para os semáforos, com as seguintes características:

1ª - Quando houver carros transitando somente na Rua B, o semáforo 2 deverá permanecer verde para que estas viaturas possam trafegar livremente.

2ª - Quando houver carros transitando somente na Rua A, o semáforo 1 deverá permanecer verde pelo mesmo motivo.

3ª - Quando houver carros transitando nas Ruas A e B, deveremos abrir o semáforo para a Rua A, pois é preferencial.

Para solucionarmos este problema, podemos utilizar um circuito lógico. Para montarmos este circuito lógico, necessitamos de sua expressão. Vamos, agora, analisando a situação, obter sua tabela da verdade.

Primeiramente, vamos estabelecer as seguintes convenções:

a) Existência de carro na Rua A: $A = 1$.

b) Não existência de carro na Rua A: $A = 0$ ou $\overline{A} = 1$.

c) Existência de carro na Rua B: $B = 1$.

d) Não existência de carro na Rua B: $B = 0$ ou $\overline{B} = 1$.

e) Verde do sinal 1 aceso: $V_1 = 1$.

f) Verde do sinal 2 aceso: $V_2 = 1$.

g) Quando $V_1 = 1 \rightarrow$ vermelho do semáforo 1 apagado: $V_{m1} = 0$,

verde do semáforo 2 apagado: $V_2 = 0$

e vermelho do semáforo 2 aceso: $V_{m2} = 1$.

h) Quando $V_2 = 1 \rightarrow V_1 = 0$, $V_{m2} = 0$ e $V_{m1} = 1$.

Vamos montar a tabela da verdade:

Situação	A	B	V_1	V_{m1}	V_2	V_{m2}
0	0	0				
1	0	1				
2	1	0				
3	1	1				

Tabela 4.1

A situação 0 (A = 0 e B = 0) representa a ausência de veículos em ambas as ruas. Se não temos carros, tanto faz qual sinal permanece aberto. Vamos adotar, por exemplo, que o verde do sinal 2 parmaneça aceso. Neste caso, preenchemos a tabela da verdade da seguinte maneira:

Situação	A	B	V_1	V_{m1}	V_2	V_{m2}
0	0	0	0	1	1	0

($V_2 = 1 \rightarrow V_1 = 0$, $V_{m1} = 1$ e $V_{m2} = 0$)

A situação 1 (A = 0 e B = 1) representa a presença de veículo na Rua B e ausência de veículo na Rua A, logo, devemos acender o sinal verde para a Rua B ($V_2 = 1$). Temos, então:

Situação	A	B	V_1	V_{m1}	V_2	V_{m2}
1	0	1	0	1	1	0

($V_2 = 1 \rightarrow V_1 = 0$, $V_{m1} = 1$ e $V_{m2} = 0$)

A situação 2 (A = 1 e B = 0) representa a presença de veículo na Rua A e ausência de veículo na Rua B, logo, devemos acender o sinal verde para a Rua A ($V_1 = 1$). Temos, então:

Situação	A	B	V_1	V_{m1}	V_2	V_{m2}
2	1	0	1	0	0	1

($V_1 = 1 \rightarrow V_2 = 0$, $V_{m2} = 1$ e $V_{m1} = 0$)

A situação 3 (A = 1 e B = 1) representa a presença de veículos em ambas as ruas, logo, devemos acender o sinal verde para a Rua A, pois esta é preferencial. Temos, então:

Situação	A	B	V_1	V_{m1}	V_2	V_{m2}
3	1	1	1	0	0	1

($V_1 = 1 \rightarrow V_{m1} = 0$, $V_2 = 0$ e $V_{m2} = 1$)

A tabela totalmente preenchida é vista a seguir:

A	B	V_1	V_{m1}	V_2	V_{m2}
0	0	0	1	1	0
0	1	0	1	1	0
1	0	1	0	0	1
1	1	1	0	0	1

Tabela 4.2

Vamos transpor a tabela para diagramas de Veitch-Karnaugh e agrupar para obtermos as expressões simplificadas das saídas V_1 , V_2 , V_{m1} e V_{m2}:

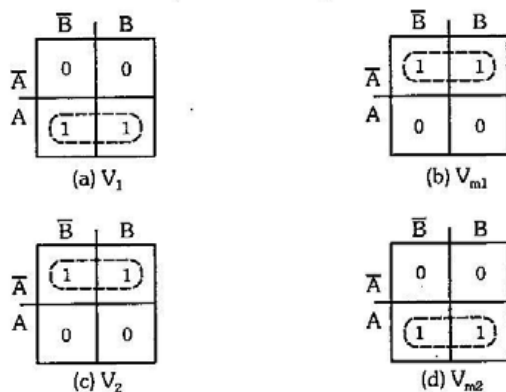



Figura 4.4

Pela tabela ou pelos diagramas, notamos que as expressões de V_1 e V_{m2} são idênticas, o mesmo ocorrendo com V_2 e V_{m1}. Assim sendo, as expressões simplificadas são:

$V_1 = V_{m2} = A$ e $V_2 = V_{m1} = \overline{A}$

O circuito, a partir destas expressões, é visto na figura 4.5

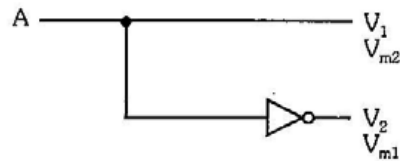



Figura 4.5

Analisando as expressões obtidas, concluímos que a presença de carros na via preferencial (A = 1) acarreta o acendimento do verde do semáforo 1 e o vermelho do 2 ($V_1 = V_{m2} = 1$) e, ainda, devido à ação do inversor no circuito, o apagamento do verde do semáforo 2 e vermelho do sinal 1 ($V_2 = V_{m1} = 0$). Da mesma forma, a ausência de carros nesta via (A = 0), causa a condição contrária ($V_1 = V_{m2} = 0$ e $V_2 = V_{m1} = 1$), que possibilita a abertura da via secundária, sendo a variável B (indicadora de veículos na Rua B) eliminada das expressões pelo processo de simplificação, pois torna-se desnecessária em função das situações consideradas no projeto.

4.2.2 Circuitos com 3 Variáveis

Deseja-se utilizar um amplificador para ligar três aparelhos: um toca-fitas, um toca-discos e um rádio FM. Vamos elaborar um circuito lógico que nos permitirá ligar os aparelhos, obedecendo às seguintes prioridades:

1ª prioridade: Toca-discos

2ª prioridade: Toca-fitas

3ª prioridade: Rádio FM

Isto significa que quando não ligarmos nem o toca-discos, nem o toca-fitas, o rádio FM, se ligado, será conectado à entrada do amplificador. Se ligarmos o toca-fitas, automaticamente o circuito conectá-lo-á à entrada do amplificador, pois possui prioridade sobre o rádio FM. Se, então, ligarmos o toca-discos, este será conectado ao amplificador, pois representa a 1ª prioridade. A partir disto, podemos montar o diagrama de blocos com as respectivas ligações:

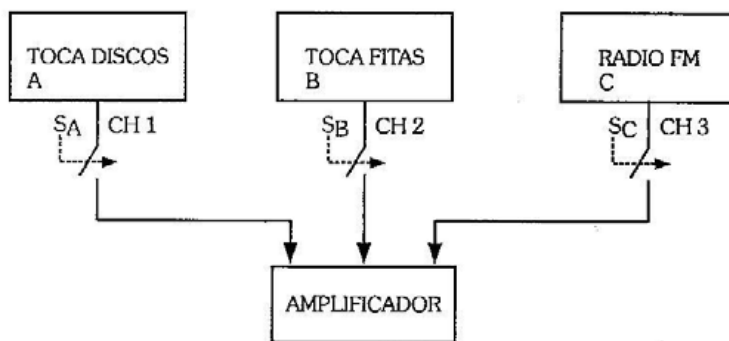



Figura 4.6

Neste projeto, o circuito lógico receberá as informações das variáveis de entrada A, B e C, representando os aparelhos, e através das saídas S_A, S_B e S_C comutará as chaves CH1, CH2 e CH3 para fazer a conexão conforme a situação requerida.

Convenções Utilizadas:

⇨ Variáveis de entrada (A, B e C): aparelho desligado = 0 e ligado = 1.

⇨ Saídas (S_A, S_B e S_C): S = 0 → chave aberta e S = 1 → chave fechada.

Tabela da Verdade

Situação	A	B	C	S_A	S_B	S_C
0	0	0	0			
1	0	0	1			
2	0	1	0			
3	0	1	1			
4	1	0	0			
5	1	0	1			
6	1	1	0			
7	1	1	1			

Tabela 4.3

Para preenchermos a tabela 4.4, vamos analisar todas as oito situações possíveis:

			S_A	S_B	S_C
Caso 0 -	Os 3 estão desligados, logo, condição irrelevante, pois não importa qual chave dever ser ligada.	⇨	X	X	X
Caso 1 -	Está ligado apenas o FM, logo somente S_C assume valor 1.	⇨	0	0	1
Caso 2 -	Está ligado apenas o toca-fitas, logo somente S_B assume valor 1.	⇨	0	1	0
Caso 3 -	Estão ligados o FM e o toca-fitas. O toca-fitas tem prioridade sobre o FM, logo somente S_B assume valor 1.	⇨	0	1	0
Caso 4 -	Está ligado apenas o toca-discos, logo somente o S_A assume o valor 1.	⇨	1	0	0
Caso 5 -	Estão ligados o toca-discos e o FM. O toca-discos é a 1ª prioridade, logo somente S_A assume valor 1.	⇨	1	0	0
Caso 6 -	Análogo ao caso 5.	⇨	1	0	0
Caso 7 -	Análogo aos casos 5 e 6.	⇨	1	0	0

Feita a análise de cada situação, podemos preencher a tabela da verdade.

Situação	A	B	C	S_A	S_B	S_C
0	0	0	0	X	X	X
1	0	0	1	0	0	1
2	0	1	0	0	1	0
3	0	1	1	0	1	0
4	1	0	0	1	0	0
5	1	0	1	1	0	0
6	1	1	0	1	0	0
7	1	1	1	1	0	0

Tabela 4.4

Transpondo para os diagramas, temos:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{A}$	X	1	0	0
A	0	0	0	0

(c) ⇒ $S_C = \overline{A}\,\overline{B}$

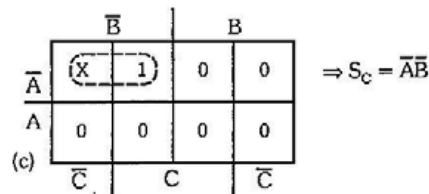
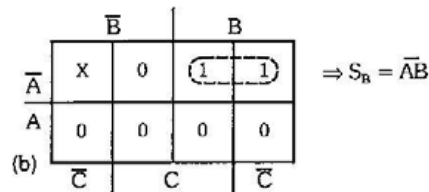
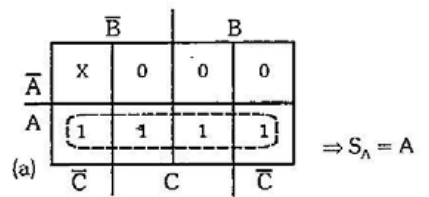



Figura 4.7

O circuito, obtido a partir das expressões, é visto na figura 4.10.

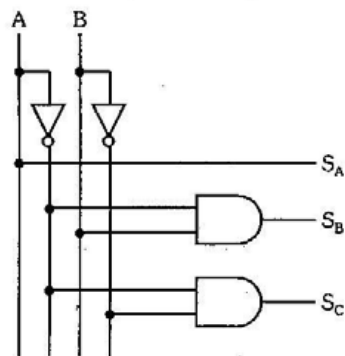



Figura 4.8

Analisando as expressões obtidas, concluímos que o toca-discos será conectado ao amplificador ($S_A = 1$ => CH1 fechada), quando for ligado ($A = 1$), independentemente dos outros aparelhos, pois $S_A = A$; que o toca-fitas será conectado ($S_B = 1$ => CH2 fechada) quando ligado ($B = 1$) e quando o toca-

discos não o for (A = 0), pois sua expressão é $S_B = \overline{A}B$ e que o rádio FM apenas será conectado (S_C = 1 => CH3 fechada), quando os outros dois não o estiverem (A = 0 e B = 0), pois $S_C = \overline{A}\,\overline{B}$. Um outro ponto importante a ser observado é que pelo fato de termos considerado a condição irrelevante do terceiro diagrama como 1 para maior simplificação, a variável C foi eliminada da expressão, bastando apenas os outros estarem desligados para que a conexão do rádio FM seja feita, sendo evidente que seu funcionamento prático, em termos de áudio, fique vinculado à sua ligação.

4.2.3 Circuitos com 4 Variáveis

Vamos supor, agora, que uma empresa queira implantar um sistema de prioridade nos seus intercomunicadores, da seguinte maneira:

Presidente: 1ª prioridade

Vice-presidente: 2ª prioridade

Engenharia: 3ª prioridade

Chefe de seção: 4ª prioridade

Esquematicamente, temos:

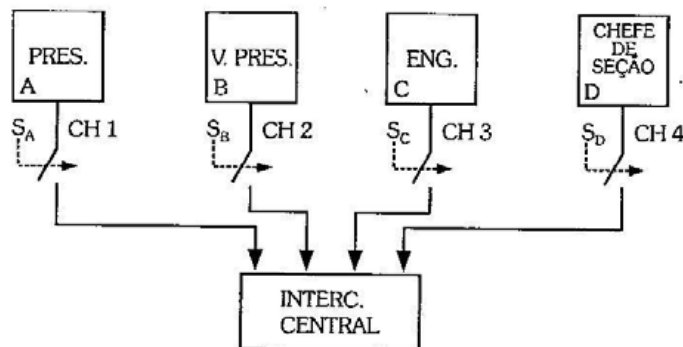



Figura 4.9

Primeiramente, vamos estabelecer as variáveis de entrada e saída do circuito lógico e as convenções do projeto:

Variáveis de entrada:

- ➪ intercomunicador do presidente: A
- ➪ intercomunicador do vice-presidente: B

⇨ intercomunicador da engenharia: C

⇨ intercomunicador do chefe de seção: D

Convenções utilizadas:

⇨ presença de chamada: 1

⇨ ausência de chamada: 0

Saídas: S_A, S_B, S_C e S_D

Convenções utilizadas:

⇨ efetivação de chamada: 1

⇨ não efetivação de chamada: 0

Estabelecidas as convenções, montamos a tabela da verdade:

A	B	C	D	S_A	S_B	S_C	S_D
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	0
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	0
1	0	0	0	1	0	0	0
1	0	0	1	1	0	0	0
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	0
1	1	1	0	1	0	0	0
1	1	1	1	1	0	0	0

Tabela 4.5

Logo após, obtemos as expressões de saída simplificadas através dos diagramas de Veitch-Karnaugh:

(a) $S_A = A$

(b) $S_B = \bar{A}B$

(c) $S_C = \bar{A}\bar{B}C$

(d) $S_D = \bar{A}\bar{B}\bar{C}D$

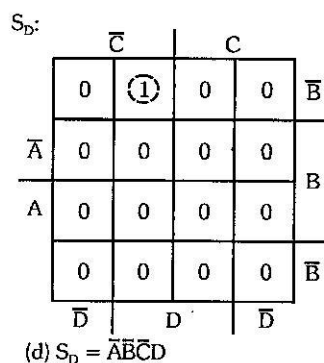
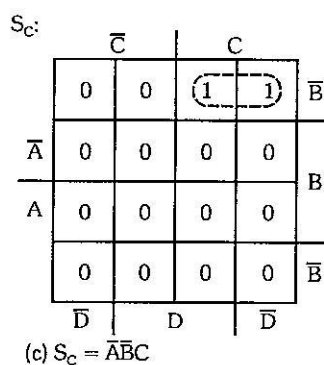
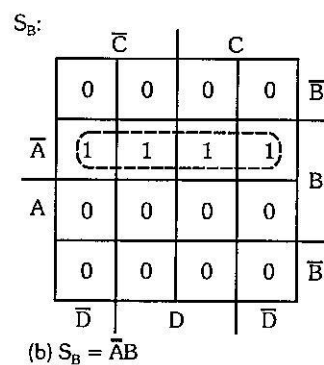
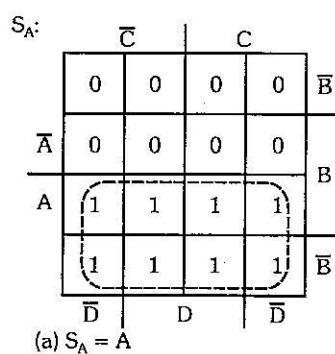



Figura 4.10

Por último, obtemos o circuito, que é visto na figura 4.11.

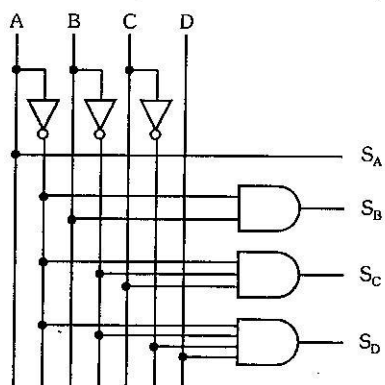



Figura 4.11

Da mesma forma que no exemplo com 3 variáveis, a saída será acionada (1) quando houver intenção de que tal situação ocorra (variável respectiva = 1) e não haja acionamento dos anteriores por ordem de prioridade (variáveis barradas = 0). Analogamente, podemos aplicar o mesmo processo para outros tipos de situações práticas que envolvam casos com prioridades, bem como, de mais variáveis.

4.2.4 Exercícios Resolvidos

1 - Elabore um circuito lógico para encher ou esvaziar um tanque industrial por meio de duas eletroválvulas, sendo uma para a entrada do líquido e outra para o escoamento de saída. O circuito lógico, através da informação de um sensor de nível máximo no tanque e de um botão interruptor de duas posições, deve atuar nas eletroválvulas para encher o tanque totalmente (botão ativado) ou, ainda, esvaziá-lo totalmente (botão desativado).

Para solucionar, vamos traçar o esquema de ligação, determinar e convencionar as variáveis de entrada e saída do circuito lógico. Este esquema é visto na figura 4.12.

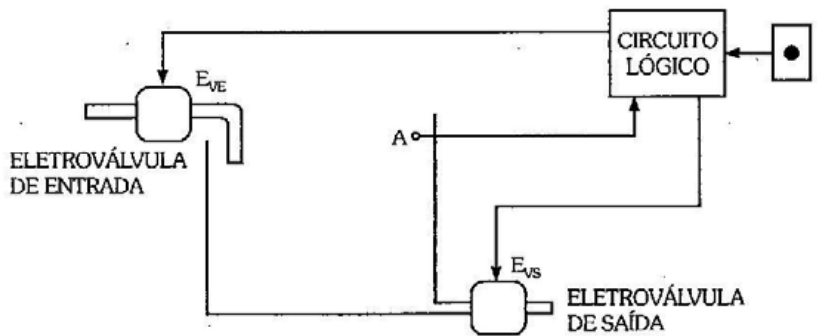



Figura 4.12

Variáveis de entrada: sensor de líquido A e botão interruptor I.

Variáveis de saída: eletroválvulas E_{VE} e E_{VS}.

Convenções:

Sensor A:

➪ presença de água = nível 1

➪ ausência de água = nível 0

Interruptor I:

- ⇨ ativado = nível 1
- ⇨ desativado = nível 0

Eletroválvulas E_{VE} e E_{VS}:

- ⇨ ligada = nível 1
- ⇨ desligada = nível 0

Em seguida, através da análise de cada caso, vamos levantar a tabela da verdade:

Caso: I = 0 e A = 0 ⇨ O caso representa o botão desativado e a ausência de líquido no sensor. O circuito não deve ligar a eletroválvula de entrada (E_{VE} = 0), mas deve ligar a eletroválvula de saída (E_{VS} = 1), para o total escoamento do líquido remanescente abaixo do nível do sensor.

Caso: I = 0 e A = 1 ⇨ O caso representa o botão desativado para esvaziamento do tanque e a presença de líquido no sensor. O circuito deve ligar apenas a eletroválvula de saída (E_{VE} = 0 e E_{VS} = 1)

Caso: I = 1 e A = 0 ⇨ Representa o botão ativado para encher o tanque, não havendo presença de líquido no sensor. O circuito deve ligar apenas a eletroválvula de entrada (E_{VE} = 1 e E_{VS} = 0)

Caso: I = 1 e A = 1 ⇨ Representa o tanque cheio e o botão ativado. Nenhuma das eletroválvulas devem ser ligadas (E_{VE} = 0 e E_{VS} = 0).

A tabela 4.6 mostra todos os casos, conforme a análise efetuada.

I	A	E_{VE}	E_{VS}
0	0	0	1
0	1	0	1
1	0	1	0
1	1	0	0

Tabela 4.6

Para simplificar a saída E_{VE}, não necessitamos do diagrama de Veitch-Karnaugh, pois teríamos apenas um termo isolado, sendo de qualquer maneira, a expressão simplificada: $E_{VE} = I\overline{A}$.

O mapa para a simplificação da saída E_{VS} é visto na figura 4.13.

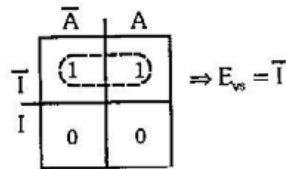



Figura 4.13

O circuito lógico obtido das expressões simplificadas é visto na figura 4.14

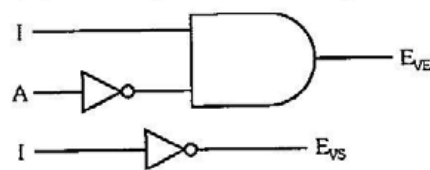



Figura 4.14

Analisando as expressões, concluímos que a eletroválvula de entrada irá funcionar (E_{VE} = 1) quando o botão interruptor estiver ativado (I = 1) e não houver a presença de líquido no sensor (A = 0), pois $E_{VE} = I\overline{A}$, e a eletroválvula de saída (E_{VS} = 1), por sua vez, apenas quando o botão interruptor não estiver ativado (I = 0), pois, $E_{VS} = \overline{I}$.

2 - Obtenha um circuito combinacional que funcione como uma chave seletora digital com 2 entradas e 1 saída digital. O circuito, em função do nível lógico aplicado a uma entrada de seleção, deve comutar à saída os sinais aplicados às entradas digitais.

Para solucionar, primeiramente, vamos esquematizar o circuito em blocos para a atribuição das variáveis de entrada e saída do sistema:

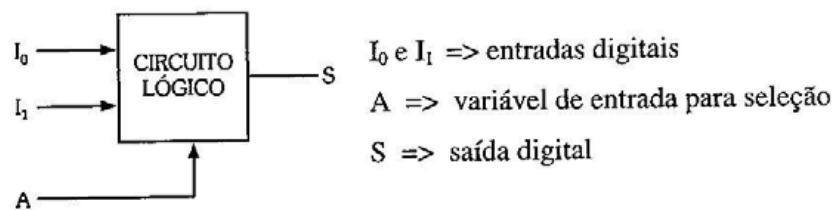



Figura 4.15

Em seguida, vamos estabelecer a convenção para a atuação da variável de seleção A:

- ➪ $A = 0 \Rightarrow I_0$ é comutado à saída S.
- ➪ $A = 1 \Rightarrow I_1$ é comutado à saída S.

Logo após, montamos a tabela da verdade colocando todas as possibilidades entre as variáveis de entrada dos níveis (I_0 e I_1), juntamente com a destinada à seleção (A):

A	I_0	I_1	S	
0	0	0	0	$S=I_0$
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	$S=I_1$
1	0	1	1	
1	1	0	0	
1	1	1	1	

Tabela 4.7

Feito isto, obtemos a expressão simplificada através do mapa de Veitch-Karnaugh visto na figura 4.16.

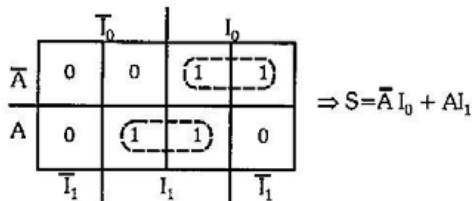



Figura 4.16

Notamos, pela expressão, que quando A = 0, o nível presente na entrada I_0 aparecerá à saída S, pois o segundo termo da expressão anular-se-á, e da mesma forma, quando A = 1, aparecerá I_1, pois o primeiro anular-se-á. A partir da expressão, desenhamos o circuito final, visto na figura 4.17.

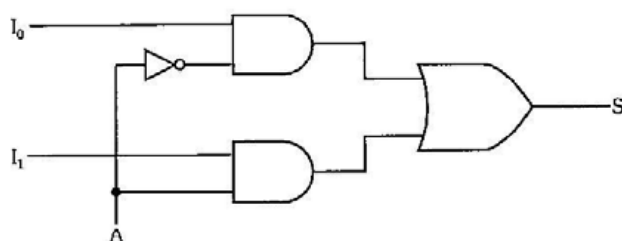



Figura 4.17

3 - Desenhe um circuito para, em um conjunto de três chaves, detectar um número par destas ligadas.

Para compensar o problema prático, principalmente da família TTL, do terminal de entrada em vazio equivaler a nível lógico 1 (veja capítulo relativo a "Família de Circuitos Lógicos"), vamos aterrar um lado das chaves, provocando no acionamento destas um nível lógico 0 no respectivo fio, ou seja, convencionar que chave fechada equivale a 0. O esquema, em blocos, é visto na figura 4.18.

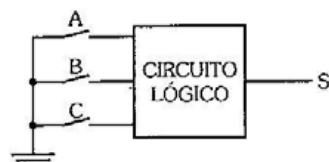



Figura 4.18

Vamos convencionar também que nos casos em que o número de chaves fechadas for par, a saída será igual a 1 (S = 1) e nos casos ímpares será 0 (S = 0).

A tabela 4.8 mostra a análise de todos os casos.

A	B	C	S	
0	0	0	0	→ 3 chaves fechadas (ímpar): S = 0
0	0	1	1	→ 2 chaves fechadas: S = 1
0	1	0	1	→ idem, S = 1
0	1	1	0	→ 1 chave fechada: S = 0
1	0	0	1	→ 2 chaves fechadas : S = 1
1	0	1	0	→ 1 chave fechada: S = 0
1	1	0	0	→ idem, S = 0
1	1	1	1	→ nenhuma chave fechada (par): S = 1

Tabela 4.8

Transpondo a tabela para o diagrama, temos:

	$\overline{B}$		B	
$\overline{A}$	0	(1)	0	(1)
A	(1)	0	(1)	0
	$\overline{C}$	C	C	$\overline{C}$

Figura 4.19

Notamos que este é um dos casos que não admitem simplificação (ver item correspondente no capítulo 3), sendo a resposta: $S = A \oplus B \oplus C$.

Através da expressão obtida, desenhamos o circuito que é visto na figura 4.20.

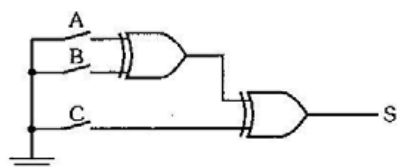



Figura 4.20

4.3 Exercícios Propostos

4.3.1 - Elabore um circuito lógico que permita encher automaticamente um filtro de água de dois recipientes e vela, conforme desenho na figura 4.21. A eletroválvula permanecerá aberta quando tivermos nível 1 de saída do circuito, e permanecerá desligada quando tivermos nível 0. O controle será efetuado por dois sensores A e B, colocados nos recipientes **a** e **b** respectivamente.

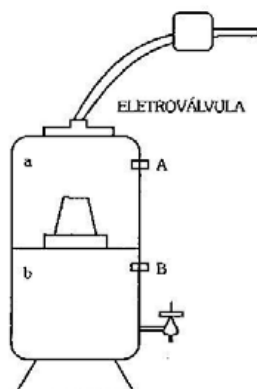



Figura 4.21

Convencionar:

⇨ recipiente vazio, sensor correspondente em nível 0.

⇨ recipiente cheio, sensor correspondente em nível 1.

4.3.2 - A figura 4.22 mostra o entroncamento das ruas A, B e C. Neste cruzamento, queremos instalar um conjunto de semáforos para as seguintes funções:

a) Quando o semáforo 1 abrir para a Rua A, automaticamente os semáforos 2 e 3 devem fechar, para possibilitar ao motorista ambas as conversões.

b) Analogamente, quando o semáforo 2 abrir, devem fechar os semáforos 1 e 3.

c) Pelo mesmo motivo, quando o semáforo 3 abrir, devem fechar os semáforos 1 e 2.

Devemos seguir também, as seguintes prioridades:

a) O motorista que está na rua A tem prioridade em relação ao motorista que está na rua B.

b) O motorista que está na rua B tem prioridade em relação ao motorista que está na rua C.

c) O motorista que está na rua C tem prioridade em relação ao motorista que está na rua A.

d) Quando houver carros nas três ruas, a rua A é preferencial.

e) Quando não houver nenhum carro nas ruas, devemos abrir o sinal para a rua A.

Obtenha as expressões e os circuitos dos sinais verdes e vermelhos, dos semáforos 1, 2 e 3.

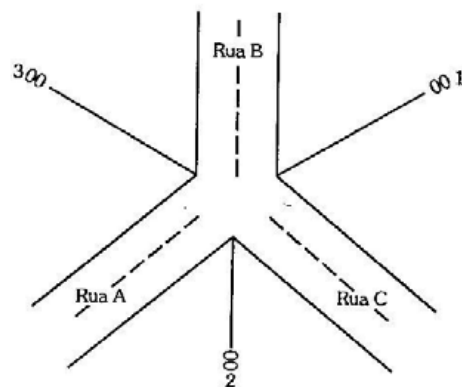



Figura 4.22

4.3.3 - Desenhe um circuito para, em um conjunto de três chaves, detectar um número ímpar destas ligadas. Convencionar que chave fechada equivale a nível 0.

4.3.4 - Estenda o projeto do exercício resolvido nº 2 para uma chave seletora digital de 4 entradas e 1 saída, sendo comutada por 2 variáveis de seleção. Desenhe o circuito completo.

4.3.5 - Projete um circuito lógico para abastecer três tanques (T1, T2 e T3) de glicose em pavimentos distintos em uma Indústria de Balas e Biscoitos, através do controle de duas bombas conforme esquematizado na figura 4.23. O abastecimento principal é feito por caminhão-tanque que fornece o produto diretamente ao T1 disposto no piso térreo localizado à entrada da empresa. Desenvolva o projeto supondo que o nível máximo de T1 seja controlado pelo caminhão, coloque os sensores de controle nas caixas, convencione as variáveis e desenhe o circuito final.

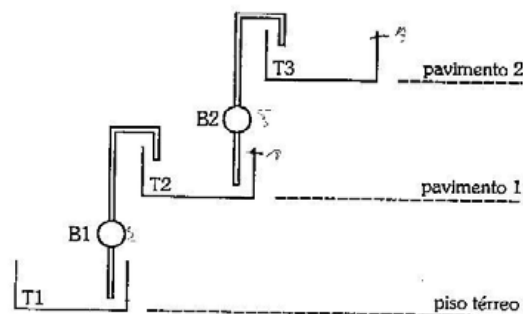
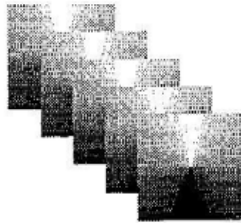



Figura 4.23

4.3.6 - Analise e faça a interpretação prática das expressões obtidas no exercício anterior.

4.3.7 - Elabore um circuito lógico para encher ou esvaziar um tanque industrial por meio de duas eletroválvulas, sendo um para a entrada do líquido e outra para o escoamento de saída. O circuito lógico, através da informação de sensores convenientemente dispostos no tanque e de um comando elétrico com dois botões interruptores, sendo cada um de duas posições, deve atuar nas eletroválvulas para encher o tanque até a metade (botão de baixo ativado), encher totalmente (ambos ativados ou apenas o de cima) ou, ainda, esvaziá-lo totalmente (botões desativados).

4.3.8 - Da mesma forma que no exercício 6, analise e faça a interpretação prática das expressões obtidas no exercício anterior.

CAPÍTULO 5

Circuitos Combinacionais 2ª Parte

5.1 Introdução

No capítulo anterior, vimos o processo de obtenção de circuitos lógicos combinacionais utilizados na solução de problemas a partir de situações práticas de maneira geral. Neste capítulo, estudaremos outros, destinados principalmente a aplicações específicas, empregados sobretudo na arquitetura interna de circuitos integrados e, ainda, em sistemas digitais.

Entre os circuitos destinados a estas finalidades destacamos os codificadores, decodificadores e os circuitos aritméticos (meio somador, somador completo, meio subtrator e subtrator completo), que serão abordados a nível básico como projetos combinacionais, para melhor entendimento, sendo entretanto encontrados na prática, disponíveis em circuitos integrados comerciais ou internos a sistemas mais complexos, tais como microprocessadores e circuitos integrados dedicados.

Para a construção dos codificadores e decodificadores, vamos inicialmente conhecer alguns códigos digitais, que serão muito úteis nos exemplos e exercícios de execução dos projetos já referidos.

5.2 Códigos

São vários os códigos dentro do campo da Eletrônica Digital, existindo situações em que a utilização de um é vantajosa em relação a outro. Vamos, neste tópico, descrever os códigos mais conhecidos.

5.2.1 Código BCD 8421

Vamos iniciar explicando que no nome deste código, a sigla BCD representa as iniciais de Binary Coded Decimal, que significa uma codificação do sistema decimal em binário. Os termos seguintes (8421) significam os valores dos algarismos num dado número binário, que conforme estudado no capítulo 1, representam respectivamente: 2^3, 2^2, 2^1 e 2^0.

A formação deste código é vista na tabela 5.1

Decimal	BCD 8421			
	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Tabela 5.1

O número de bits de um código é o número de dígitos binários que este possui. Notamos, então, que o código BCD 8421 é um código de 4 bits e, ainda, que é válido de 0 a 9_{10}.

5.2.2 Outros Códigos BCD de 4 Bits

Existem vários outros, dentre os quais vamos destacar o BCD 7421, BCD 5211 e BCD 2421.

A regra de conversão destes códigos para o sistema decimal é análoga à vista para o BCD 8421. As formações destes códigos são mostradas na tabela 5.2.

Decimal	BCD 7421	BCD 5211	BCD 2421
0	0000	0000	0000
1	0001	0001	0001
2	0010	0011	0010
3	0011	0101	0011
4	0100	0111	0100
5	0101	1000	1011
6	0110	1001	1100
7	1000	1011	1101
8	1001	1101	1110
9	1010	1111	1111

Tabela 5.2

5.2.3 Código Excesso 3

Este nada mais é do que a transformação do número decimal no binário correspondente, somando-se 3 unidades.

Exemplo: $0_{10} = 0000 \rightarrow$ somando-se 3 unidades, temos: 0011.

A formação do código é vista na tabela 5.3.

Decimal	Excesso 3			
	A	B	C	D
0	0	0	1	1
1	0	1	0	0
2	0	1	0	1
3	0	1	1	0
4	0	1	1	1
5	1	0	0	0
6	1	0	0	1
7	1	0	1	0
8	1	0	1	1
9	1	1	0	0

Tabela 5.3

Este código é utilizado em alguns casos nos Circuitos Aritméticos.

5.2.4 Código Gray

Sua principal característica é que de um número a outro apenas um bit varia. Sua formação é mostrada na tabela 5.4.

Decimal	Gray			
	A	B	C	D
0	0	0	0	0
1	0	0	0	1
2	0	0	1	1
3	0	0	1	0
4	0	1	1	0
5	0	1	1	1
6	0	1	0	1
7	0	1	0	0
8	1	1	0	0
9	1	1	0	1
10	1	1	1	1
11	1	1	1	0
12	1	0	1	0
13	1	0	1	1
14	1	0	0	1
15	1	0	0	0

Tabela 5.4

O código Gray, transpondo para o Diagrama de Veitch-Karnaugh, apresenta a seguinte ordem de colocação:

Figura 5.1

5.2.5 Códigos de 5 Bits

Destacaremos apenas os dois mais importantes:

1) Código 2 entre 5

Trata-se de um código que possui sempre 2 bits iguais a 1, dentro de 5 bits. Sua formação é vista na tabela 5.5.

Decimal	2 entre 5				
	A	B	C	D	E
0	0	0	0	1	1
1	0	0	1	0	1
2	0	0	1	1	0
3	0	1	0	0	1
4	0	1	0	1	0
5	0	1	1	0	0
6	1	0	0	0	1
7	1	0	0	1	0
8	1	0	1	0	0
9	1	1	0	0	0

Tabela 5.5

2) Código Johnson

Trata-se de um código que será utilizado na construção do Contador Johnson. Sua formação é vista na tabela 5.6.

Decimal	Johnson				
	A	B	C	D	E
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	1
3	0	0	1	1	1
4	0	1	1	1	1
5	1	1	1	1	1
6	1	1	1	1	0
7	1	1	1	0	0
8	1	1	0	0	0
9	1	0	0	0	0

Tabela 5.6

5.2.6 Código 9876543210

Este código de 10 bits foi bastante utilizado na época em que os sistemas mostradores de algarismos eram válvulas eletrônicas (Nixie e Numitron). Algumas dessas válvulas possuíam cada algarismo composto por uma placa ou filamento, arranjado apropriadamente no formato do número.

Notamos no código, que em 10 saídas somente uma vale 1 em cada caso, acendendo assim o algarismo correspondente. A formação deste código é vista na tabela 5.7.

Decimal	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	0	1	0	0
3	0	0	0	0	0	0	1	0	0	0
4	0	0	0	0	0	1	0	0	0	0
5	0	0	0	0	1	0	0	0	0	0
6	0	0	0	1	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0

Tabela 5.7

5.3 Codificadores e Decodificadores

Vamos, agora, tratar de circuitos que efetuam a passagem de um determinado código para outro. Primeiramente, vamos fazer uma análise do significado das palavras codificador e decodificador.

Chamamos de codificador o circuito combinacional que torna possível a passagem de um código conhecido para um desconhecido. Como exemplo, podemos citar o circuito inicial de uma calculadora que transforma uma entrada decimal, através do sistema de chaves de um teclado, em saída binária para que o circuito interno processe e faça a operação.

Chamamos de decodificador o circuito que faz o inverso, ou seja, passa um código desconhecido para um conhecido. No exemplo citado é o circuito que recebe o resultado da operação em binário e o transforma em saída decimal, na forma compatível para um mostrador digital apresentar os algarismos.

A figura 5.2 ilustra o exemplo utilizado.

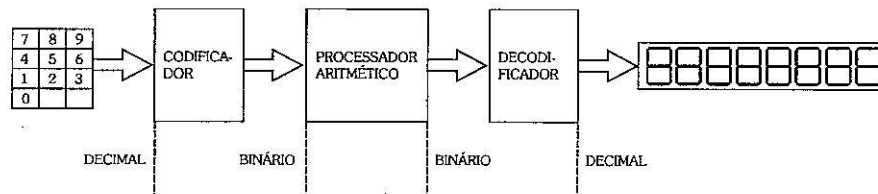



Figura 5.2

Os termos codificador e decodificador, porém, diferenciam-se em função do referencial. Se para o usuário da calculadora o sistema de entrada é um codificador, para o processador será um decodificador, pois passa de um código desconhecido para ele (decimal), para um conhecido (binário). Na prática, é comum se utilizar a denominação de decodificador para o sistema que passa de um código para outro, quaisquer que sejam.

5.3.1 Codificador Decimal/Binário

Vamos, neste item, elaborar um codificador para transformar um código decimal em binário (BCD8421). A entrada do código decimal vai ser feita através de um conjunto de chaves numeradas de 0 a 9 e a saída por 4 fios, para fornecer um código binário de 4 bits, correspondente à chave acionada. A figura 5.3 mostra a estrutura geral deste sistema, sendo convencionado que a chave fechada equivale a nível 0, para evitar o problema prático, principalmente da família TTL (ver capítulo 9), que um terminal de entrada em vazio é equivalente a nível lógico1.

Figura 5.3

A seguir, vamos construir a tabela da verdade do codificador que relaciona cada chave de entrada decimal com a respectiva saída em binário:

Chave	A	B	C	D
Ch0	0	0	0	0
Ch1	0	0	0	1
Ch2	0	0	1	0
Ch3	0	0	1	1
Ch4	0	1	0	0
Ch5	0	1	0	1
Ch6	0	1	1	0
Ch7	0	1	1	1
Ch8	1	0	0	0
Ch9	1	0	0	1

Tabela 5.8

Através da tabela, concluímos que a saída A valerá 1 quando Ch8 ou Ch9 for acionada. A saída B quando Ch4, Ch5, Ch6 ou Ch7 for acionada. A saída C quando Ch2, Ch3, Ch6 ou Ch7 for acionada. A saída D quando Ch1, Ch3, Ch5, Ch7 ou Ch9 for acionada.

Usaremos para a construção do circuito, uma porta NE em cada saída, pois esta fornece nível 1 quando qualquer uma de suas entradas assumir nível 0, situação compatível com a convenção adotada para o conjunto de chaves. A ligação das entradas de cada porta será feita, conforme a análise efetuada, às chaves responsáveis pelos níveis 1 de cada saída.

O circuito, assim constituído, é visto na figura 5.4.

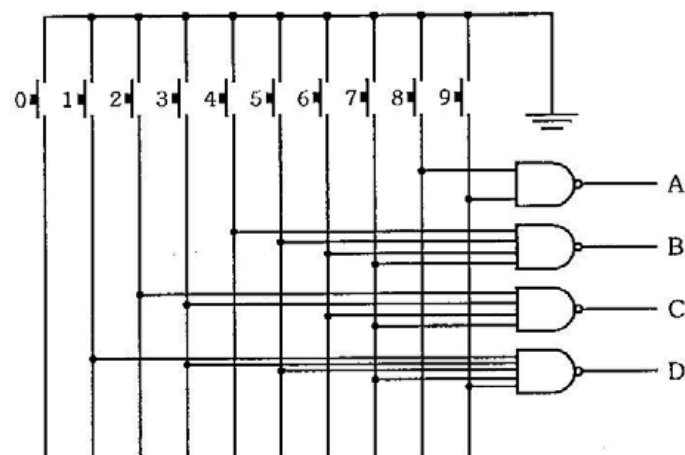



Figura 5.4

Pela figura, notamos que a chave Ch0 não está ligada a nenhuma das entradas das portas, sendo irrelevante o seu acionamento, pois a saída também será igual a 0 (A = B = C = D = 0) quando nenhuma das chaves for acionada.

5.3.2 Decodificador Binário/Decimal

A estrutura geral deste decodificador é vista na figura 5.5.

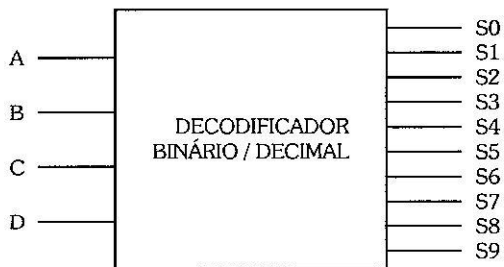



Figura 5.5

Vamos montar a tabela da verdade do circuito no qual as entradas são bits do código BCD 8421 e as saídas são os respectivos bits do código decimal 9876543210.

BCD 8421				Código 9876543210									
A	**B**	**C**	**D**	**S9**	**S8**	**S7**	**S6**	**S5**	**S4**	**S3**	**S2**	**S1**	**S0**
0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	0	0	0
0	1	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	1	0	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	0	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0

Tabela 5.9

O código BCD 8421 não possui números maiores que 9, logo, tanto faz o valor assumido nas possibilidades excedentes, visto que, quando passarmos do código BCD 8421 para o código 9876543210 estas não irão ocorrer. Nos diagramas da Veitch-Karnaugh, conseqüentemente, consideraremos estes casos

como condições irrelevantes. A figura 5.6 mostra os diagramas de todas as saídas do decodificador (S_9 a S_0) e suas respectivas simplificações.

S_9:

(a) $S_9 = AD$

S_8:

(b) $S_8 = A\overline{D}$

S_7:

(c) $S_7 = BCD$

S_6:

(d) $S_6 = BC\overline{D}$

S_5:

(e) $S_5 = B\overline{C}D$

S_4:

(f) $S_4 = B\overline{C}\overline{D}$

S_3:

(g) $S_3 = \overline{B}CD$

S_2:

(h) $S_2 = \overline{B}C\overline{D}$

S_1:

(i) $S_1 = \overline{A}\,\overline{B}\,\overline{C}D$

S_0:

(j) $S_0 = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$

Figura 5.6

A partir das expressões simplificadas, obtemos o circuito do decodificador que é visto na figura 5.7.

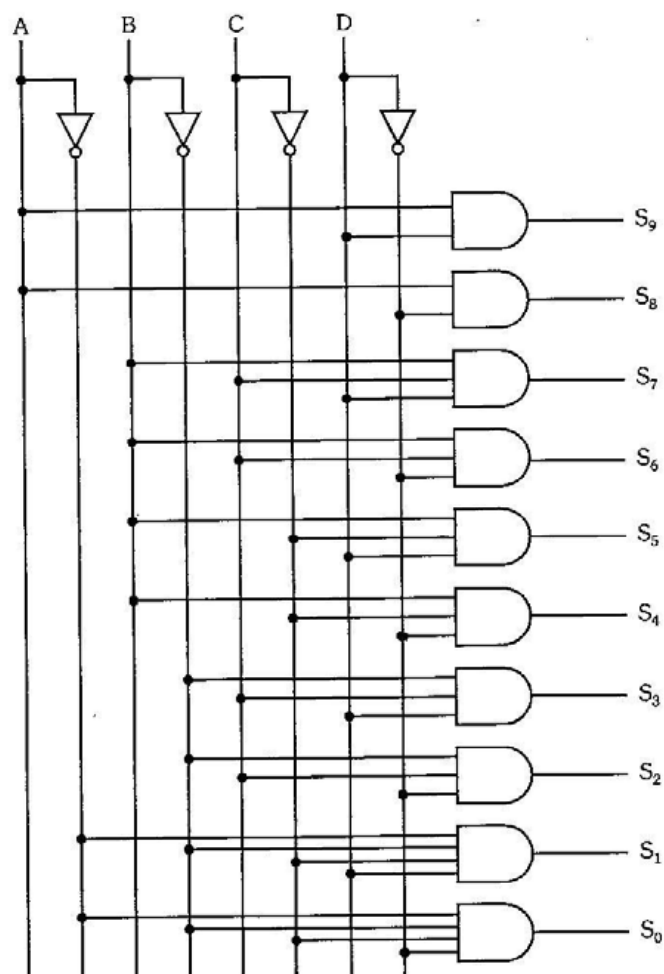



Figura 5.7

5.3.3 Projetos de Decodificadores

Agindo de forma análoga ao processo visto no decodificador Binário/Decimal, podemos construir decodificadores que passem de qualquer código para qualquer outro. Para isso, basta montarmos a tabela da verdade, simplificar as expressões de saída e implementarmos o circuito.

Para exemplificar, vamos elaborar o decodificador de BCD 8421 para Excesso 3. Inicialmente, montamos a tabela da verdade:

BCD 8421				Excesso 3			
A	B	C	D	S_3	S_2	S_1	S_0
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

Tabela 5.10

Podemos notar que o código BCD 8421 é utilizado para representar até o algarismo 9. As outras possibilidades não irão ocorrer, logo, para estas condições a resposta torna-se irrelevante.

Para simplificar as expressões, vamos montar os diagramas de Veitch-Karnaugh.

S_3:

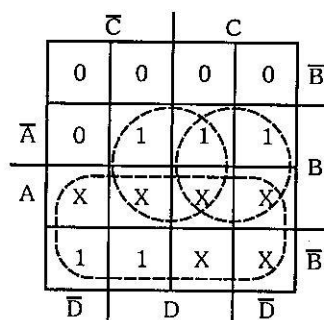



Figura 5.8

Agrupamentos: 1 oitava A e 2 quadras BD e BC,

$\therefore S_3 = A + BD + BC$

S_2:

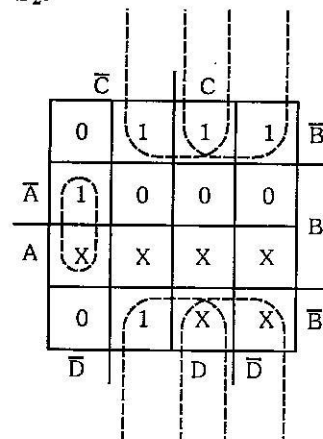



Figura 5.9

Agrupamentos: 2 quadras $\overline{B}D$, $\overline{B}C$ e 1 par $B\overline{C}\overline{D}$,

$\therefore S_2 = \overline{B}D + \overline{B}C + B\overline{C}\overline{D}$

S_1:

	$\overline{C}\,\overline{D}$	$\overline{C}D$	CD	$C\overline{D}$	
	1	0	1	0	$\overline{B}$
$\overline{A}$	1	0	1	0	B
A	X	X	X	X	B
	1	0	X	X	$\overline{B}$

Figura 5.10

Agrupamentos: 2 quadras $\overline{C}\,\overline{D}$ e CD.

$\therefore S_1 = \overline{C}\,\overline{D} + CD$ ou $S_1 = C \odot D$

S_0:

Figura 5.11

Agrupamento: 1 oitava $\overline{D}$.

$\therefore S_0 = \overline{D}$

O circuito decodificador, obtido a partir das expressões, é visto na figura 5.12.

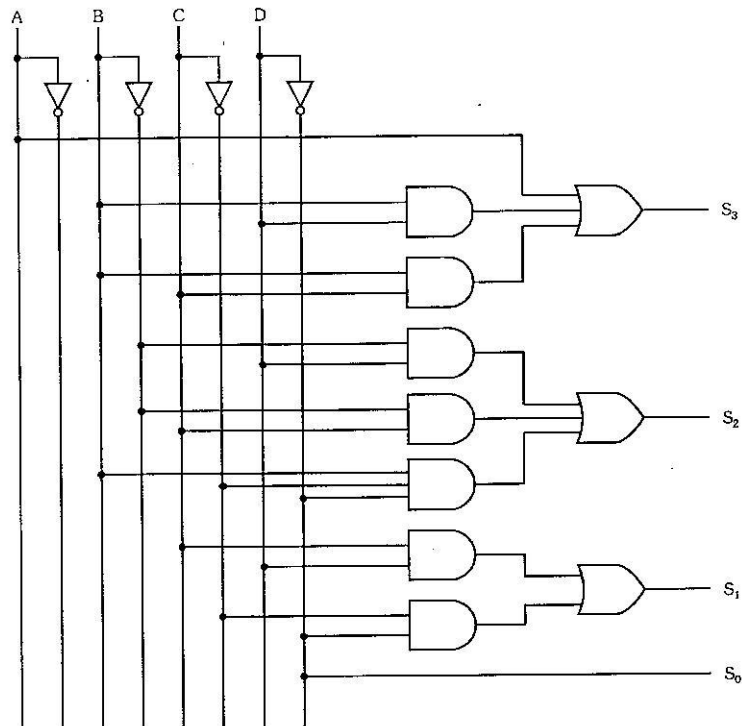



Figura 5.12

No circuito, ao ser aplicado o código BCD 8421 nos terminais de entrada A, B, C e D, teremos nos terminais de saída S_3, S_2, S_1 e S_0, o código Excesso 3.

Vamos, a seguir, para exemplificar a utilização de um código diferente de BCD 8421 na entrada, elaborar o decodificador inverso, ou seja, que transforme do código Excesso 3 para BCD 8421.

Agindo da mesma maneira, montamos a tabela da verdade:

Excesso 3				BCD 8421			
A	**B**	**C**	**D**	S_8	S_4	S_2	S_1
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1

Tabela 5.11

Da mesma forma, os casos não existentes serão considerados como irrelevantes.

Vamos simplificar estas saídas mediante a utilização dos diagramas de Veitch-Karnaugh. Na colocação, devemos achar a região indicada pela possibilidade assumida pela entrada, e, nesta região, colocar o valor assumido pela saída, pois, neste caso, o código de entrada é o Excesso 3, não sendo válida a ordem de colocação, já vista, para o BCD 8421.

Transpondo as saídas para os diagramas, temos:

Figura 5.13

Agrupamentos: 1 quadra AB e
1 par ACD.

$\therefore S_8 = AB + ACD$

S_4:

Figura 5.14

Agrupamentos: 2 quadras $\overline{B}\overline{D}$ e $\overline{B}\overline{C}$ e 1 par BCD.

$\therefore S_4 = \overline{B}\overline{D} + \overline{B}\overline{C} + BCD$

S_2:

Figura 5.15

Agrupamento: 2 quadras $\overline{C}D$ e $C\overline{D}$

$\therefore S_2 = \overline{C}D + C\overline{D}$ ou $S_2 = C \oplus D$

S_1:

Figura 5.16

Agrupamento: 1 oitava $\overline{D}$,

$\therefore S_1 = \overline{D}$

O circuito deste decodificador é visto na figura 5.17.

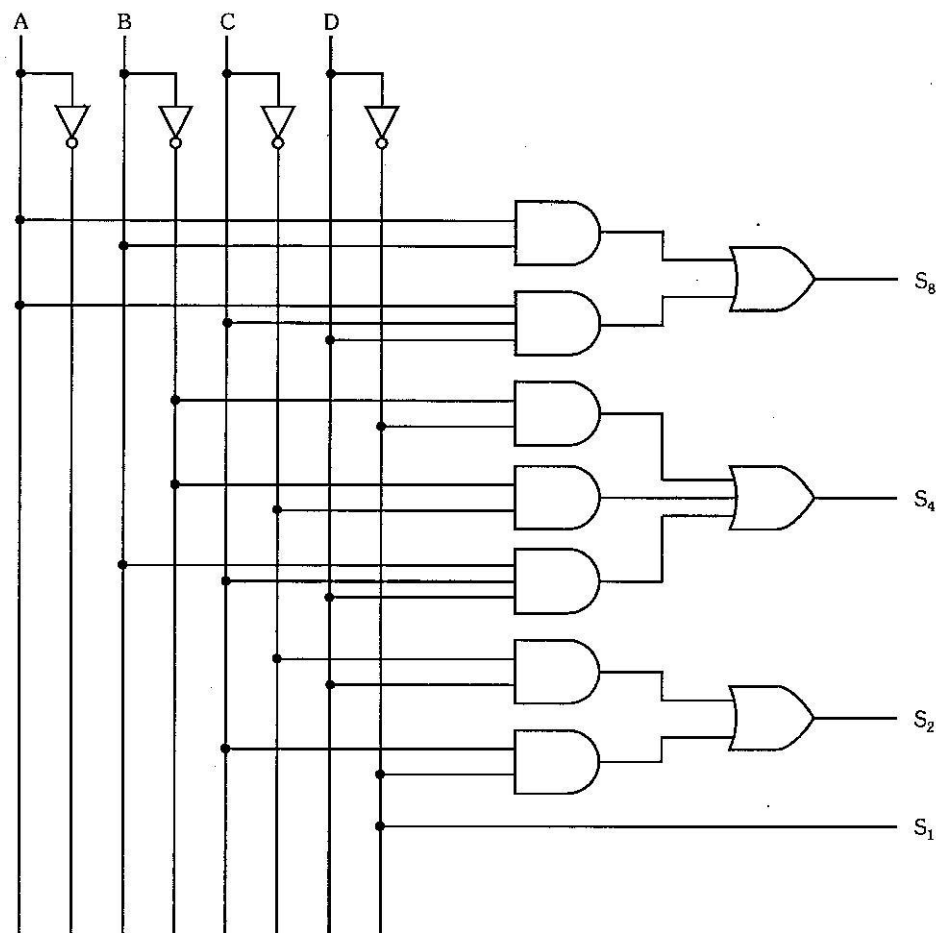



Figura 5.17

Se no circuito, aplicarmos nas entradas A, B, C e D, o código Excesso 3, teremos nas saídas S_8, S_4, S_2 e S_1, o código BCD 8421.

5.3.4 Decodificador para Display de 7 Segmentos

O **display de 7 segmentos** possibilita escrevermos números decimais de 0 a 9 e alguns outros símbolos que podem ser letras ou sinais. A figura 5.8 representa uma unidade do display genérica, com a nomenclatura de identificação dos segmentos usual em manuais práticos.

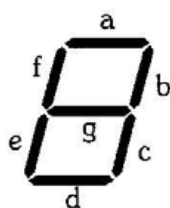



Figura 5.18

Entre as tecnologias de fabricação das unidades de display usaremos o mais comum que é o display a led, que possui cada segmento composto por um led, existindo um tipo denominado **catodo comum** e outro **anodo comum.**

O display tipo catodo comum é aquele que possui todos os catodos dos led's interligados, sendo necessário aplicar nível 1 no anodo respectivo para acender cada segmento. Já o de anodo comum possui todos os anodos interligados, sendo preciso aplicar nível 0 ao catodo respectivo.

Vamos a título de exemplo, elaborar um decodificador para a partir de um código binário (BCD 8421) escrever a seqüência de 0 a 9 em um display de 7 segmentos catodo comum. O esquema geral deste decodificador é visto na figura 5.19.

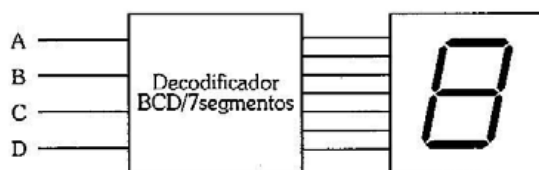



Figura 5.19

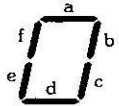
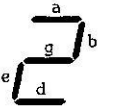
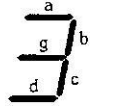
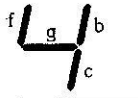
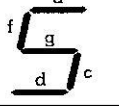
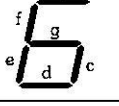
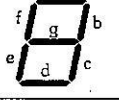
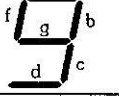
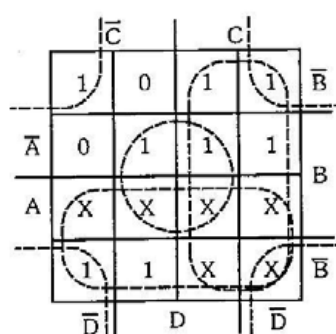
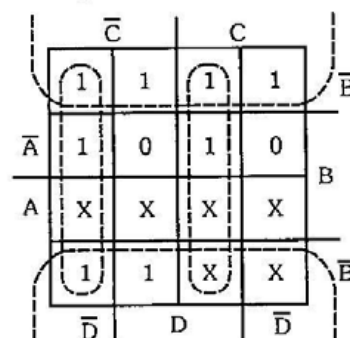
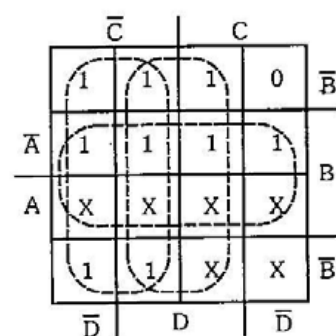
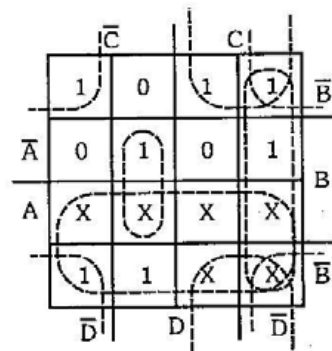
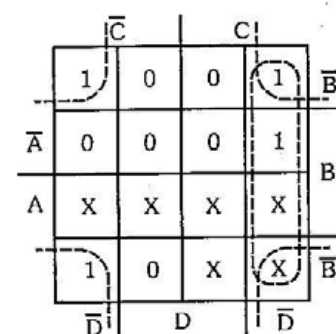
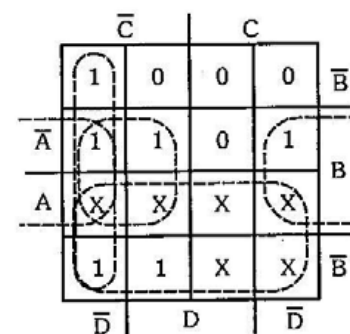
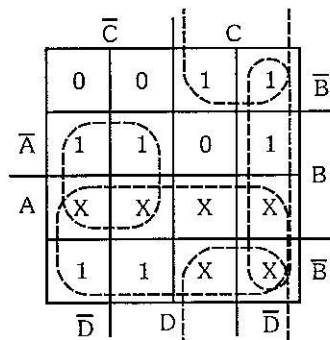
Para efetuar o projeto deste decodificador, devemos verificar em cada caractere os segmentos que devem ser acesos e atribuir o nível 1 (no caso do catodo comum), em função da respectiva entrada no código binário. A tabela 5.12 apresenta a seqüência de caracteres, o respectivo código de entrada e os níveis aplicados em cada segmento para que tal ocorra.

Characteres	Display	BCD 8421				Código para 7 Segmentos						
		A	B	C	D	a	b	c	d	e	f	g
0		0	0	0	0	1	1	1	1	1	1	0
1		0	0	0	1	0	1	1	0	0	0	0
2		0	0	1	0	1	1	0	1	1	0	1
3		0	0	1	1	1	1	1	1	0	0	1
4		0	1	0	0	0	1	1	0	0	1	1
5		0	1	0	1	1	0	1	1	0	1	1
6		0	1	1	0	1	0	1	1	1	1	1
7		0	1	1	1	1	1	1	0	0	0	0
8		1	0	0	0	1	1	1	1	1	1	1
9		1	0	0	1	1	1	1	1	0	1	1

Tabela 5.12

Para fins de simplificação, vamos considerar os casos fora da seqüência como irrelevantes. Transpondo as saídas para os diagramas, temos:

(a) $a = A + C + BD + \overline{B}\,\overline{D}$

ou $a = A + C + B \odot D$

(b) $b = \overline{B} + \overline{C}\,\overline{D} + CD$

ou $b = \overline{B} + C \odot D$

(c) $c = B + \overline{C} + D$

(d) $d = A + \overline{B}\,\overline{D} + \overline{B}C + C\overline{D} + B\overline{C}D$

(e) $e = \overline{B}\,\overline{D} + C\overline{D}$

(f) $f = A + \overline{C}\,\overline{D} + B\overline{C} + B\overline{D}$

(g) $g = A + B\overline{C} + \overline{B}C + C\overline{D}$

ou $g = A + B \oplus C + C\overline{D}$

Figura 5.20

O circuito do decodificador BCD 8421 para display de 7 segmentos obtido, é visto na figura 5.21 na página seguinte.

Convém observar que o circuito poderia ser otimizado, pois as expressões dos segmentos possuem vários termos em comum, resultando no emprego de um menor número de portas. Porém, para melhor clareza didática, este foi deixado na sua forma original de acordo com as expressões extraídas dos diagramas.

Um outro ponto a ser realçado é que numa montagem prática, a ligação do display se faz, conforme a família lógica, através de resistores para observar os limites máximos de corrente nos led's, ou ainda, utilizando outras estratégias para controlar o brilho, como por exemplo, blocos open-collector (ver capítulo relativo à "Famílias de Circuitos Lógicos").

Os displays de 7 segmentos podem ainda escrever outros caracteres, que são freqüentemente utilizados em sistemas digitais para representar outras funções, bem como formar palavras-chave em software de programação. A tabela 5.13 mostra como exemplo, outras possibilidades de caracteres.

A	b	C / c	d	E / e	F
G / g	H / h	I / i	J	L	N
n	O / o	P	q	r	S
t	u	U	y	-	⊐

Tabela 5.13

Para efetuar o projeto, basta verificar caso a caso quais segmentos devem acender e montar assim, a tabela da verdade.

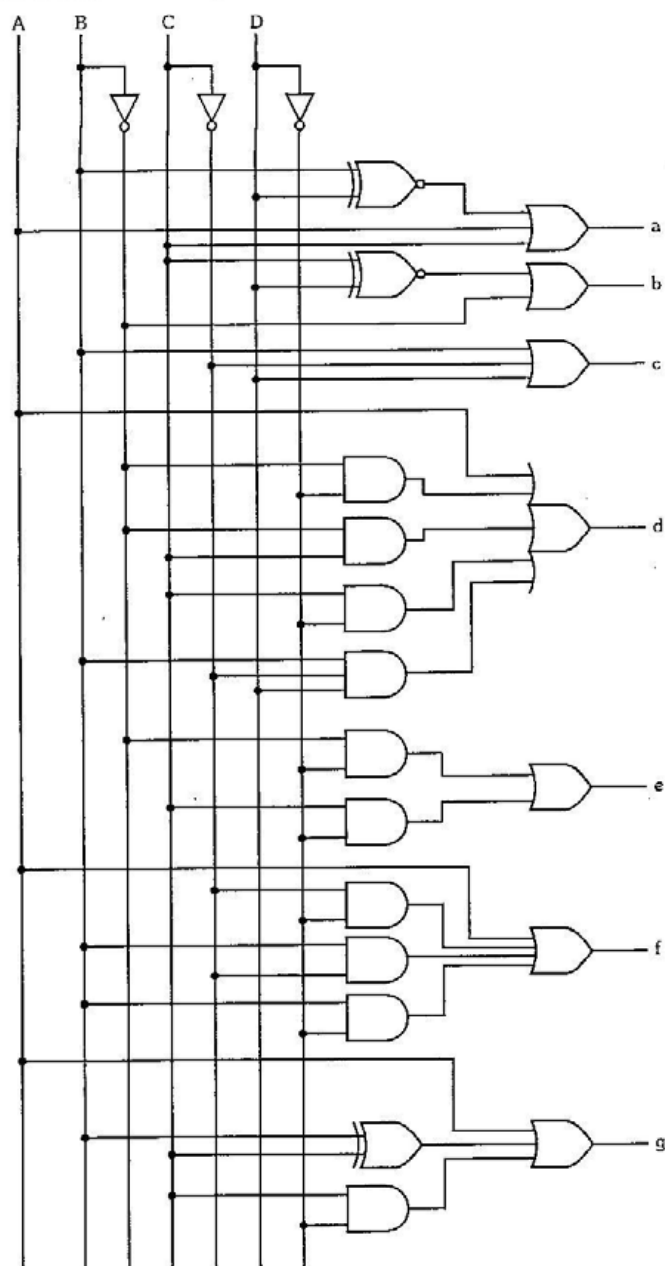



Figura 5.21

5.3.5 Exercícios Resolvidos

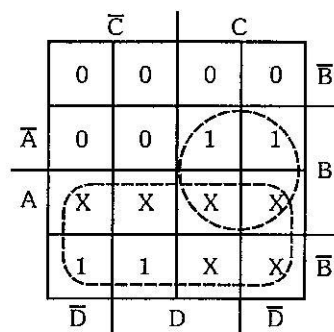
1 - Elabore o decodificador BCD 8421 para 2 entre 5.

Agimos de maneira análoga aos exemplos anteriores, montando primeiramente a tabela da verdade:

BCD 8421				2 entre 5				
A	B	C	D	S_4	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	1	1
0	0	0	1	0	0	1	0	1
0	0	1	0	0	0	1	1	0
0	0	1	1	0	1	0	0	1
0	1	0	0	0	1	0	1	0
0	1	0	1	0	1	1	0	0
0	1	1	0	1	0	0	0	1
0	1	1	1	1	0	0	1	0
1	0	0	0	1	0	1	0	0
1	0	0	1	1	1	0	0	0

Tabela 5.14

Da mesma forma, vamos considerar as saídas não existentes como condições irrelevantes. Transpondo as saídas para os diagramas, temos:

S_4:

(a) $S_4 = A + BC$

S_3:

(b) $S_3 = B\overline{C} + AD + \overline{B}CD$

S_2:

(c) $S_2 = A\overline{D} + \overline{B}C\overline{D} + \overline{A}\,\overline{C}D$

S_1:

(d) $S_1 = \overline{A}\,\overline{B}\,\overline{D} + \overline{A}\,\overline{C}\,\overline{D} + BCD$

S_0:

(e) $S_0 = \overline{A}\,\overline{B}\,\overline{C} + BC\overline{D} + \overline{B}CD$

Figura 5.22

O circuito obtido a partir das expressões é visto na figura 5.23.

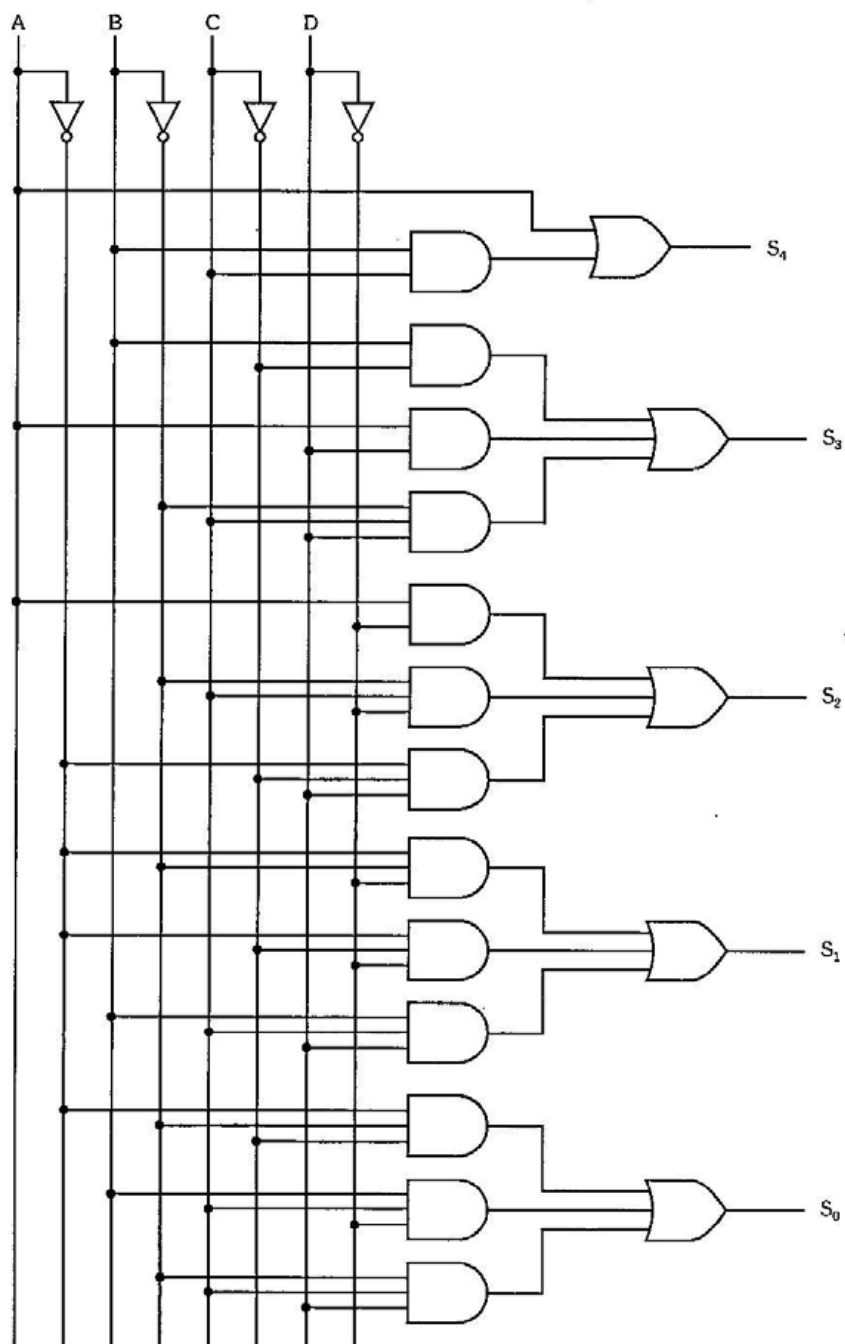



Figura 5.23

2 - Projete um decodificador que transforme do código Gray para o sistema binário comum.

Vamos montar a tabela da verdade:

Código Gray				Binário			
A	B	C	D	S_3	S_2	S_1	S_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	1	0	0	1	0
0	0	1	0	0	0	1	1
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
0	1	0	1	0	1	1	0
0	1	0	0	0	1	1	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	0	1	0
1	1	1	0	1	0	1	1
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	0
1	0	0	0	1	1	1	1

Tabela 5.15

Devemos lembrar aqui que a colocação no diagrama de Veitch-Karnaugh se faz de acordo com a possibilidade assumida pelas variáveis de entrada, sendo neste caso o código Gray. Feita esta observação, vamos, então, simplificar as expressões:

S_3:

(a) $S_3 = A$

S_2:

(b) $S_2 = \bar{A}B + A\bar{B}$

ou $S_2 = A \oplus B$

Figura 5.24

S_1:

$$S_1 = \bar{A}\bar{B}C + \bar{A}B\bar{C} + A\bar{B}\bar{C} + ABC$$

Figura 5.25

Fatorando a expressão, temos: $S_1 = \bar{A}\overbrace{\left(\bar{B}C + B\bar{C}\right)}^{X} + A\overbrace{\left(BC + \bar{B}\bar{C}\right)}^{\bar{X}}$

Lembrando que: $\bar{X}Y + X\bar{Y} = X \oplus Y$, podemos escrever:

$S_1 = \bar{A}X + A\bar{X} = A \oplus X \quad \therefore \quad S_1 = A \oplus B \oplus C$

S_0:

Figura 5.26

$S_0 = A \oplus B \oplus C \oplus D$ (ver capítulo 3: "casos que não admitem simplificação").

A partir destas expressões, obtemos o circuito final:

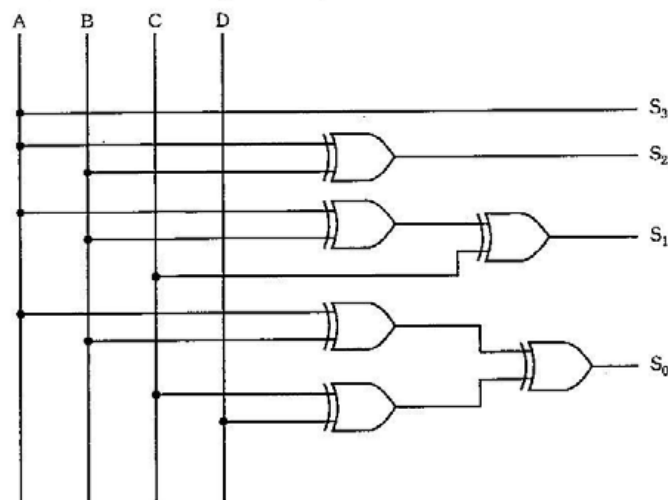



Figura 5.27

3 - Projete um decodificador para, a partir de um código binário, escrever a seqüência da figura 5.28 em um display de 7 segmentos catodo comum.

CARACTERE	S	t	o	P	-	E	r	8
CASO	0	1	2	3	4	5	6	7

Figura 5.28

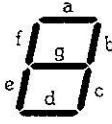
Para escrever os 8 símbolos mostrados na figura, um código binário de 3 bits é suficiente. A tabela 5.16 apresenta o código binário de entrada e os níveis aplicados em cada segmento para escrever a seqüência de caracteres.

	A	B	C	a	b	c	d	e	f	g
	0	0	0	1	0	1	1	0	1	1
	0	0	1	0	0	0	1	1	1	1
a	0	1	0	0	0	1	1	1	0	1
f g b	0	1	1	1	1	0	0	1	1	1
e d c	1	0	0	0	0	0	0	0	0	1
	1	0	1	1	0	0	1	1	1	1
	1	1	0	0	0	0	0	1	0	1
	1	1	1	1	1	1	1	1	1	1

Tabela 5.16

A figura 5.29 apresenta o diagrama e a simplificação do circuito de saída para cada segmento.

a:

(a) $\overline{A}\overline{B}\overline{C} + BC + AC$

b:

(b) $b = BC$

c:

(c) $c = \overline{A}\overline{C} + ABC$

d:

(d) $d = \overline{A}\overline{C} + AC + \overline{B}C$ ou

$d = A \odot C + \overline{B}C$

(e) $e = B + C$ (f) $f = \overline{A}\,\overline{B} + C$ (g) $g = 1$

Figura 5.29

O circuito extraído das expressões simplificadas é visto na figura 5.30.

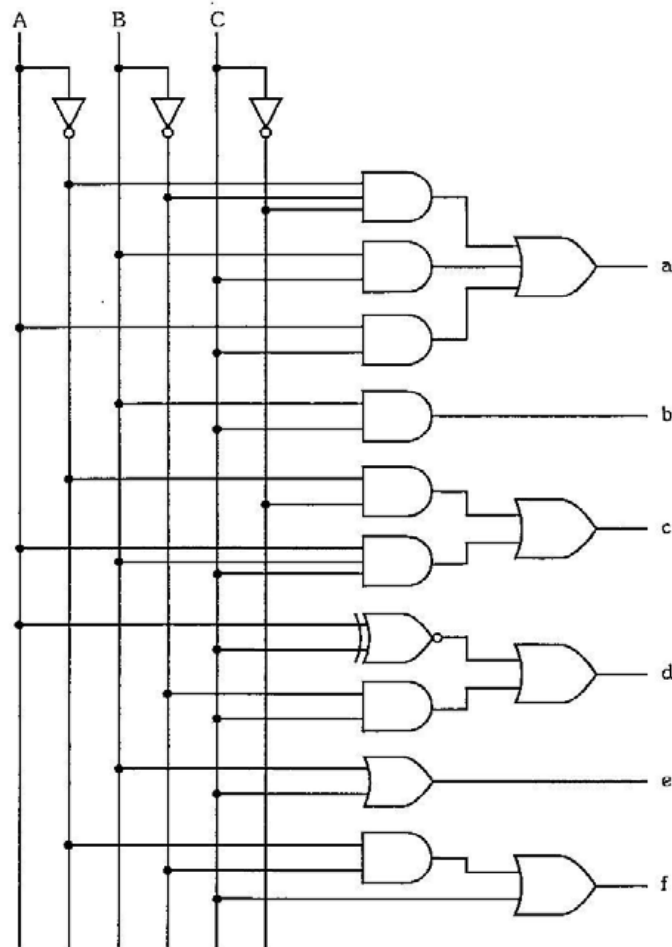



Figura 5.30

Conforme mostra a simplificação, o segmento g irá permanecerá aceso em todos os casos, pois g = 1. No circuito, no caso de uma montagem prática, essa ligação deverá ser feita através de um resistor, convenientemente calculado conforme o Vcc, para não danificar o display.

5.4 Circuitos Aritméticos

Dentro do conjunto de circuitos combinacionais aplicados para finalidade específica nos sistemas digitais, destacam-se os circuitos aritméticos. São utilizados, principalmente, para construir a **ULA (Unidade Lógica Aritmética)** dos microprocessadores e, ainda, encontrados disponíveis em circuitos integrados comerciais. Neste tópico, abordaremos os principais circuitos aritméticos e seus subsistemas derivados.

5.4.1 Meio Somador

Antes de iniciarmos o assunto, vamos relembrar alguns tópicos importantes da soma de 2 números binários:

$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ +\ 1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +\ 0 \\ \hline 1 \end{array} \qquad \begin{array}{r} {}^{1}1 \\ +\ 1 \\ \hline 10 \end{array}$$

transporte

Após essa breve introdução, vamos montar uma tabela da verdade da soma de 2 números binários de 1 algarismo:

A	B	S	Ts
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tabela 5.17

Ts → transporte de saída

(0 + 0 = 0 → Ts = 0)

(0 + 1 = 1 → Ts = 0)

(1 + 0 = 1 → Ts = 0)

(1 + 1 = 0 → Ts = 1)

Representando cada número por 1 bit, podemos, então, montar um circuito que possui como entradas A e B, e como saída, a soma dos algarismos (S) e o respectivo transporte de saída (Ts). As expressões características do circuito, extraídas da tabela, são:

$S = A \oplus B$

$Ts = AB$

O circuito a partir destas expressões é visto na figura 5.31.

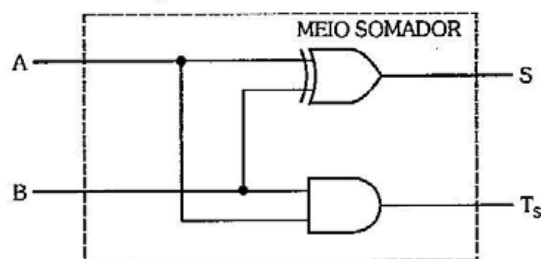



Figura 5.31

A representação em bloco deste circuito é vista na figura 5.32.

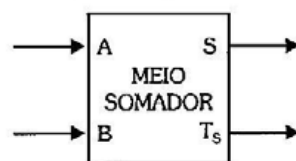
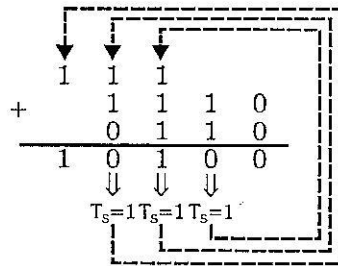



Figura 5.32

Este circuito Meio Somador é também conhecido como **Half Adder**, sendo a saída de transporte denominada **carry out**, ambos os termos derivados do inglês.

5.4.2 Somador Completo

O Meio Somador possibilita efetuar a soma de números binários com 1 algarismo. Para se fazer a soma de números binários de mais algarismos, esse circuito torna-se insuficiente, pois não possibilita a introdução do transporte de entrada proveniente da coluna anterior. Para melhor compreensão, vamos analisar o caso da soma: $1110_2 + 110_2$. Assim sendo, temos:

A coluna 1 tem como resultado um transporte de saída igual a 0. A coluna 2 tem como resultado 0 e um transporte de saída igual a 1. A coluna 3 tem um transporte de entrada igual a 1 (Ts da coluna anterior), possui resultado 1 e transporte de saída igual a 1. A coluna 4 tem transporte de entrada igual a 1, resultado 0 e transporte de saída 1. A coluna 5 possui apenas um transporte de entrada (Ts da coluna 4) e, obviamente, seu resultado será igual a 1.

Para fazermos a soma de 2 números binários de mais algarismos, basta somarmos coluna a coluna, levando em conta o transporte de entrada que nada mais é do que o Ts da coluna anterior.

O Somador Completo é um circuito para efetuar a soma completa de uma coluna, considerando o transporte de entrada. Vamos, agora, montar a tabela da verdade deste circuito:

A	B	T_E	S	T_S
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tabela 5.18

$T_E \rightarrow$ transporte de entrada

$(0 + 0 + 0 = 0 \rightarrow Ts = 0)$

$(0 + 0 + 1 = 1 \rightarrow Ts = 0)$

$(0 + 1 + 0 = 1 \rightarrow Ts = 0)$

$(0 + 1 + 1 = 0 \rightarrow Ts = 1)$

$(1 + 0 + 0 = 1 \rightarrow Ts = 0)$

$(1 + 0 + 1 = 0 \rightarrow Ts = 1)$

$(1 + 1 + 0 = 0 \rightarrow Ts = 1)$

$(1 + 1 + 1 = 1 \rightarrow Ts = 1)$

Vamos, então, escrever as expressões características, sem simplificação, de um Somador Completo:

$$S = \overline{A}\,\overline{B}T_E + \overline{A}B\overline{T}_E + A\overline{B}\,\overline{T}_E + ABT_E$$

$$Ts = \overline{A}BT_E + A\overline{B}T_E + AB\overline{T}_E + ABT_E$$

Transpondo para diagramas de Veitch-Karnaugh, temos:

S:

Figura 5.33

Conforme já estudado, podemos escrever:

$S = A \oplus B \oplus T_E$

T_S:

$T_S = BT_E + AT_E + AB$

Figura 5.34

Vamos, através das expressões, esquematizar o circuito Somador Completo:

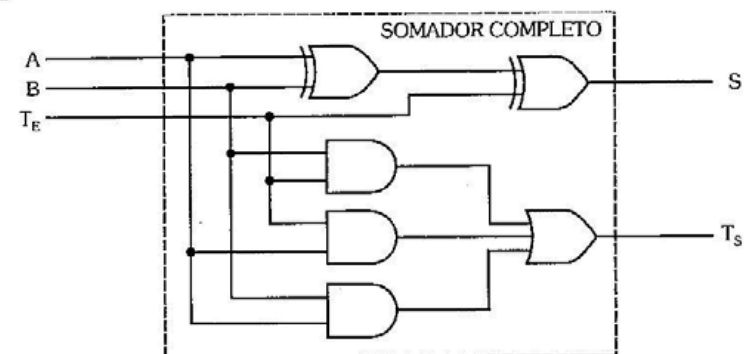



Figura 5.35

Da mesma forma, o circuito apresentado em bloco, é visto na figura 5.36.

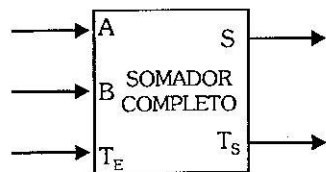



Figura 5.36

O circuito Somador Completo é também conhecido como **Full Adder**, sendo a entrada de transporte denominada **carry in**, ambos os termos derivados do inglês.

Vamos, para exemplo de aplicação, montar um sistema em blocos que efetua a soma de 2 números de 4 bits, conforme o esquema a seguir:

$$\begin{array}{ccccc} & A_3 & A_2 & A_1 & A_0 \\ + & B_3 & B_2 & B_1 & B_0 \\ \hline S_4 & S_3 & S_2 & S_1 & S_0 \end{array}$$

Para efetuar a soma dos bits A_0 e B_0 dos números (1ª coluna), vamos utilizar um Meio Somador, pois não existe transporte de entrada, mas para as outras colunas utilizaremos Somadores Completos, pois necessitaremos considerar os transportes provenientes das colunas anteriores. O sistema montado é visto na figura 5.37.

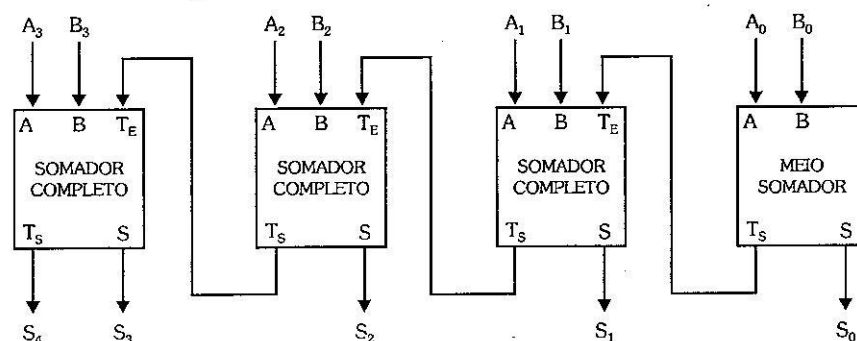



Figura 5.37

Generalizando para um sistema que efetua a soma de 2 números de m bits (m = n +1), temos:

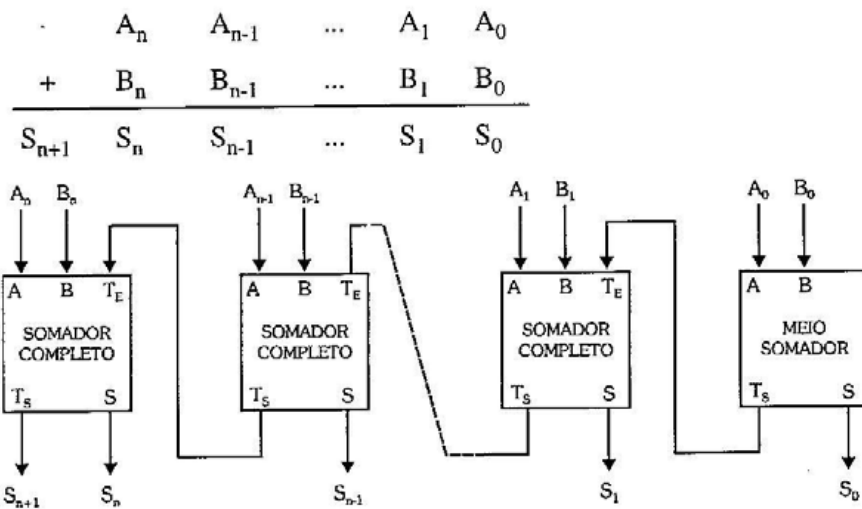



Figura 5.38

5.4.3 Somador Completo a partir de Meio Somadores

Podemos construir um Somador Completo a partir de 2 Meio Somadores. Para isso, vamos analisar as expressões de ambos os blocos:

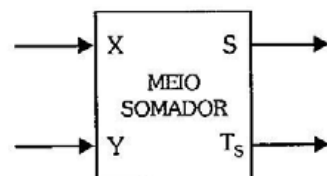
Meio Somador:

$$S = X \oplus Y$$
$$T_s = XY$$

Figura 5.39

Somador Completo:

$$S = A \oplus B \oplus T_E$$
$$Ts = \overline{A}BT_E + A\overline{B}T_E + AB\overline{T}_E + ABT_E$$

Figura 5.40

Fatorando a expressão de Ts, temos:

$$Ts = T_E\ (\overline{A}B + A\overline{B}) + AB\ (\overline{T_E} + T_E)\ \therefore\ Ts = T_E\ (A \oplus B) + AB$$

Ligando A e B nas entradas do Meio Somador 1, temos:

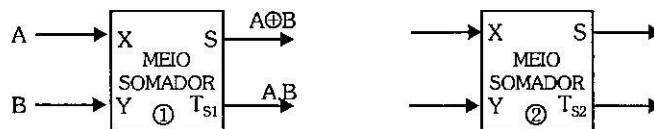



Figura 5.41

Ligando a saída S do Meio Somador 1 à entrada X do outro Meio Somador e à entrada Y deste, a variável T_E, temos:

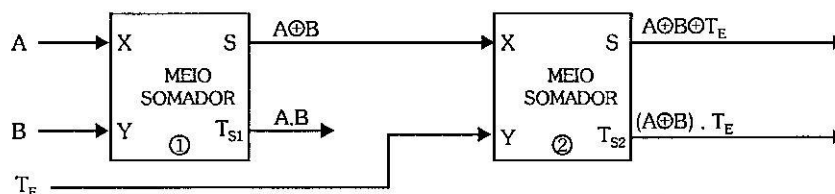



Figura 5.42

Notamos que a saída S do Meio Somador 2 apresenta a soma completa de 2 números.

Analisando as saídas T_{S1} e T_{S2}, notamos que são os termos da expressão de Ts de um Somador Completo, logo se fizermos a soma dessas 2 saídas (Porta OU), teremos na saída o Ts de um Somador Completo. A figura 5.43 mostra o circuito completo com essa ligação.

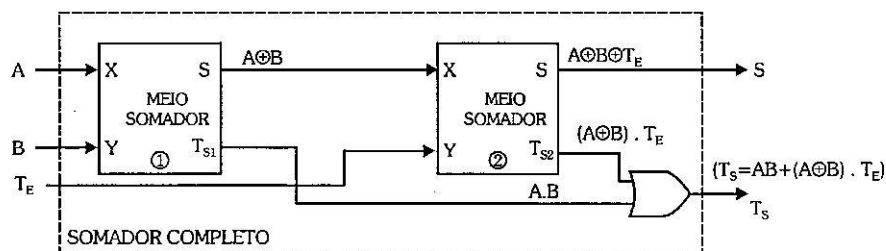



Figura 5.43

5.4.4 Meio Subtrator

Antes de iniciarmos o assunto, vamos relembrar alguns tópicos importantes da subtração de números binários:

0 - 0 = 0

0 - 1 = 1 e transporta 1 ("empresta" 1)

1 - 0 = 1

1 - 1 = 0

Vamos montar a tabela da verdade de uma subtração de 2 números binários de 1 algarismo:

A	B	S	Ts
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

(0 - 0 = 0 → Ts = 0)

(0 - 1 = 1 → Ts = 1)

(1 - 0 = 1 → Ts = 0)

(1 - 1 = 0 → Ts = 0)

Tabela 5.19

Representando cada número por 1 bit, podemos montar um circuito com as entradas A e B, e como saída, a subtração (S) e o transporte de saída (Ts).

As expressões características do circuito, extraídas da tabela, são:

$S = A \oplus B$

$Ts = \overline{A}B$

O circuito a partir destas, é visto na figura 5.44.

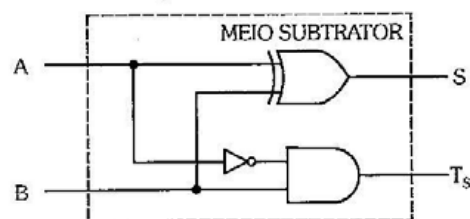



Figura 5.44

Em bloco, o circuito recebe a representação da figura 5.45.

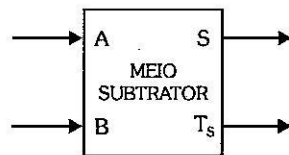



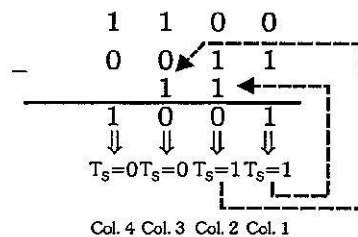
Figura 5.45

Do inglês, o circuito recebe a denominação **Half Subractor.**

5.4.5 Subtrator Completo

O Meio Subtrator possibilita-nos efetuar a subtração de números binários de 1 algarismo. Para se fazer uma subtração com números de mais algarismos, este circuito torna-se insuficiente, pois não possibilita a entrada do transporte (T_E), proveniente da coluna anterior.

Para compreendermos melhor, vamos analisar a subtração:

$1100_2 - 11_2$. Assim sendo, temos:

A coluna 1 tem como resultado de saída 1 e apresenta um transporte de saída igual a 1. A coluna 2 tem um transporte de entrada igual a 1 (Ts da coluna anterior), um resultado igual a 0 e um Ts = 1. A coluna 3 tem: $T_E = 1$, resultado igual a 0 e Ts = 0 . A coluna 4 tem: $T_E = 0$, resultado igual a 1 e Ts = 0.

Para fazermos a subtração de números binários de mais algarismos, basta subtrairmos coluna a coluna, levando em conta o transporte de entrada, que nada mais é do que o Ts da coluna anterior.

O Subtrator Completo é um circuito que efetua a subtração completa de uma coluna, ou seja, considera o transporte de entrada proveniente da coluna anterior. Vamos, agora, montar a tabela da verdade deste circuito:

A	B	T_E	S	T_S
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Tabela 5.20

As expressões características extraídas da tabela são:

$$S = \overline{A}\,\overline{B}T_E + \overline{A}B\overline{T}_E + A\overline{B}\overline{T}_E + ABT_E$$

$$Ts = \overline{A}\,\overline{B}T_E + \overline{A}B\overline{T}_E + \overline{A}BT_E + ABT_E$$

Vamos simplificar estas expressões:

S:

T_S:

(a) $S = A \oplus B \oplus T_E$

(b) $Ts = \overline{A}B + \overline{A}T_E + BT_E$

Figura 5.46

O circuito derivado das expressões é visto na figura 5.47.

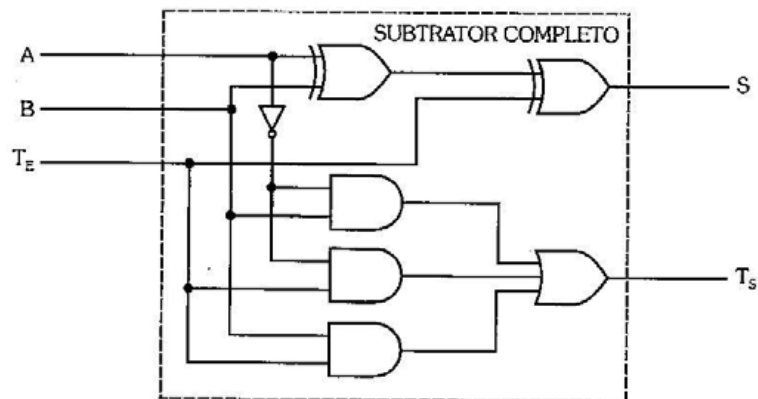



Figura 5.47

Em bloco, recebe a representação da figura 5.48.

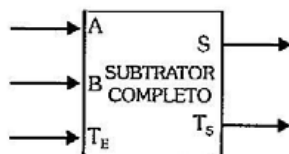



Figura 5.48

A denominação derivada do inglês é **Full Subtractor**.

Da mesma forma, podemos esquematizar um sistema subtrator para 2 números de m bits (m = n + 1). A figura 5.49 mostra um sistema subtrator genérico para 2 números de m bits.

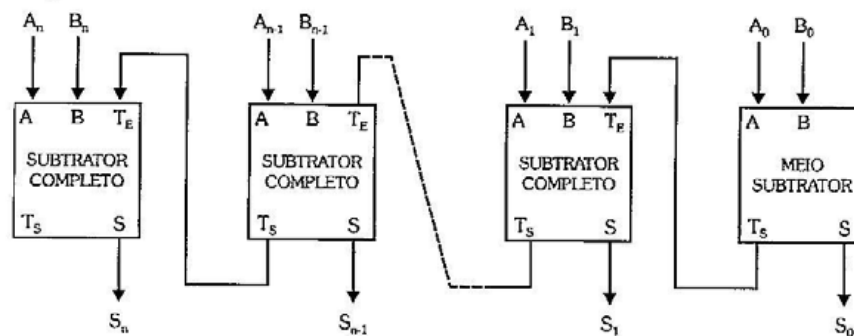



Figura 5.49

Neste sistema, a saída de transporte (Ts) do último bloco torna-se desnecessária se o número $A_n...A_0$ (minuendo) for maior ou igual a $B_n...B_0$

(subtraendo), porém poderá ser utilizada no caso contrário para sinalizar que o resultado é negativo, estando, então, na notação do complemento de 2.

5.4.6 Subtrator Completo a partir de Meio Subtratores

Podemos construir um Subtrator Completo a partir de 2 Meio Subtratores. Para isso, vamos analisar as expressões de ambos os blocos:

Meio Subtrator:

$$S = X \oplus Y$$
$$T_s = \overline{X}Y$$

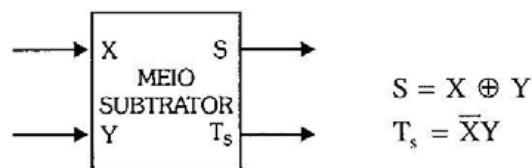



Figura 5.50

Subtrator Completo:

$$S = A \oplus B \oplus T_E$$
$$Ts = \overline{A}\,\overline{B}T_E + \overline{A}B\overline{T}_E + \overline{A}BT_E + ABT_E$$

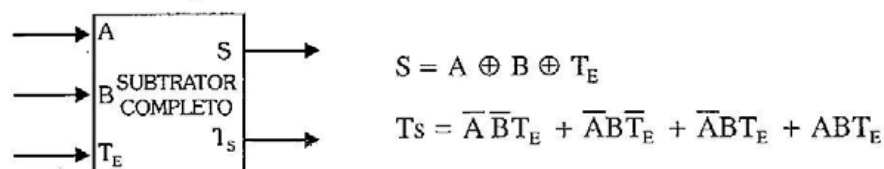



Figura 5.51

Fatorando a expressão de Ts, temos:

$$Ts = T_E\ (\overline{A}\,\overline{B} + AB) + \overline{A}B(\overline{T}_E + T_E)$$

$$Ts = T_E\,(A \odot B) + \overline{A}B \quad \therefore \quad Ts = T_E\,(\overline{A \oplus B}) + \overline{A}B$$

Ligando A e B nas entradas X e Y do Meio Subtrator 1, temos:

Figura 5.52

Ligando a saída S na entrada X do 2º bloco, e à entrada Y, a variável T_E, temos:

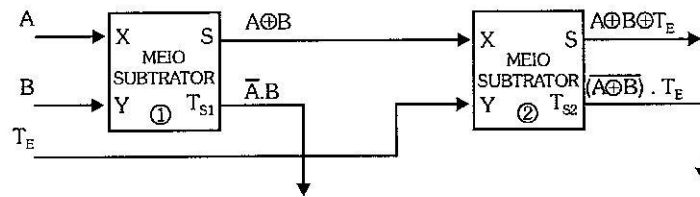



Figura 5.53

Notamos que a saída S do Meio Subtrator 2 apresenta a subtração completa de 2 números.

Analisando as saídas T_{S1} e T_{S2}, notamos que são os termos da expressão de Ts de um Subtrator Completo. Se injetarmos T_{S1} e T_{S2} nas entradas de uma porta OU, teremos na saída o Ts de um Subtrator Completo. O circuito com essa ligação é visto na figura 5.54.

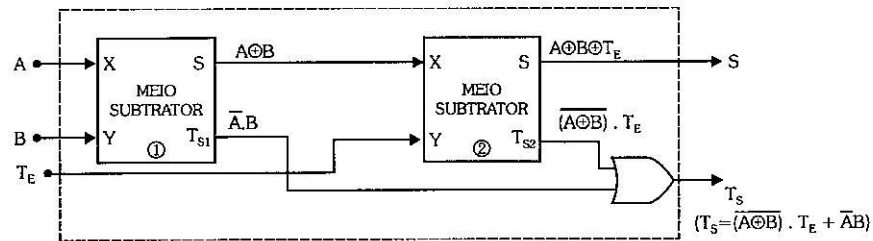



Figura 5.54

5.4.7 Somador/Subtrator Completo

Podemos esquematizar um circuito que efetue as duas operações. Para isso, vamos introduzir uma outra entrada que permanecendo em nível 0, faz o circuito efetuar uma soma completa, e permanecendo em nível 1, faz efetuar uma subtração completa.

Vamos, agora, montar a tabela da verdade do circuito, sendo M a variável de controle (M = 0 → soma e M = 1 → subtração):

M	A	B	T_E	S	T_S
0	0	0	0	0	0
0	0	0	1	1	0
0	0	1	0	1	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	0	1
0	1	1	0	0	1
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	1	1	1
1	0	1	0	1	1
1	0	1	1	0	1
1	1	0	0	1	0
1	1	0	1	0	0
1	1	1	0	0	0
1	1	1	1	1	1

Soma Completa (M = 0)

Subtração Completa (M = 1)

Tabela 5.21

Vamos simplificar as saídas S e Ts, através dos diagramas de Veitch-Karnaugh:

S:

Figura 5.55

Do diagrama, obtemos:

$$S = A\overline{B}\,\overline{T}_E + \overline{A}\,\overline{B}T_E + ABT_E + \overline{A}B\overline{T}_E$$

Fatorando a expressão, temos:

$$S = \overline{A}(\overline{B}T_E + B\overline{T}_E) + A(\overline{B}\,\overline{T}_E + BT_E)$$

$$S = \overline{A}(B \oplus T_E) + A(B \odot T_E)$$

$$S = \overline{A}(B \oplus T_E) + A(\overline{B \oplus T_E})$$

$$\therefore S = A \oplus B \oplus T_E$$

Ts:

Figura 5.56

Do diagrama, obtemos: $Ts = BT_E + \overline{M}AB + \overline{M}AT_E + M\overline{A}B + M\overline{A}T_E$

Fatorando a expressão, temos:

$$Ts = BT_E + B(\overline{M}A + M\overline{A}) + T_E(M\overline{A} + \overline{M}A)$$

$$Ts = BT_E + B(M \oplus A) + T_E(M \oplus A)$$

$$Ts = BT_E + (M \oplus A)(B + T_E)$$

Vamos, então, esquematizar o circuito:

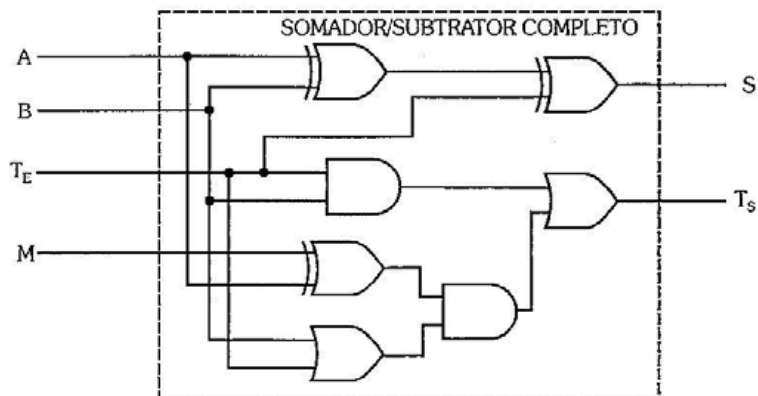



Figura 5.57

A figura 5.58 mostra a representação deste circuito Somador/Subtrator Completo, em bloco:

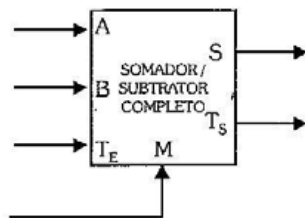



Figura 5.58

5.4.8 Exercícios Resolvidos

1 - Desenhe um sistema somador para 2 números de 2 bits apenas com blocos de Somadores Completos.

Para obtermos este sistema, necessitaríamos de um Meio Somador e um Somador Completo. A solução é obtida aplicando nível 0 (terra) à entrada de transporte do (T_E) do somador relativo ao bit menos significativo, transformando-o em Meio Somador, pois esta entrada fica eliminada. A figura 5.59 apresenta este sistema, composto apenas de Somadores Completos.

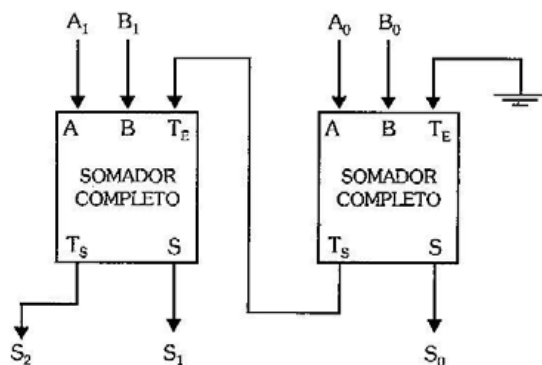



Figura 5.59

2 - Desenvolva um circuito com uma entrada de controle M, para fornecer à saída o complemento de 1 de um número binário de 1 bit. (M = 0 = > Saída = número de entrada e M = 1 = > Saída = complemento de 1).

Para solucionar, vamos levantar a tabela da verdade, considerando a variável de controle M.

M	A	S	
0	0	0	Saída = número de entrada
0	1	1	
1	0	1	Saída = complemento de 1
1	1	0	

Tabela 5.22

A partir da tabela, obtemos a expressão: $S = \overline{M}A + M\overline{A}$ ou $S = M \oplus A$, sendo o circuito derivado, visto na figura 5.60.

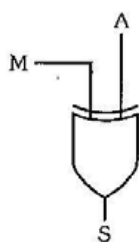



Figura 5.60

Através do circuito, podemos constatar que M igual a 0 a saída é igual ao bit A da entrada ($A = 0 \Rightarrow 0 \oplus 0 = 0$ e $A = 1 \Rightarrow 0 \oplus 1 = 1$), e para M igual a 1 a saída é oposta ($A = 0 \Rightarrow 1 \oplus 0 = 1$ e $A = 1 \Rightarrow 1 \oplus 1 = 0$).

3 - Esquematize, em blocos, um sistema subtrator para 2 números com 2 bits.

O sistema proposto irá realizar a subtração do número A_1A_0 com o número B_1B_0. Assim sendo, temos:

$$\begin{array}{rcc} & A_1 & A_0 \\ - & B_1 & B_0 \\ \hline & S_1 & S_0 \end{array}$$

Para a 1ª coluna da operação, vamos utilizar um Meio Subtrator, pois não há transporte de entrada. Para a 2ª coluna, porém, utilizamos um Subtrator Completo, pois este possui entrada para o bit proveniente da coluna anterior. O circuito, assim esquematizado, é visto na figura 5.61.

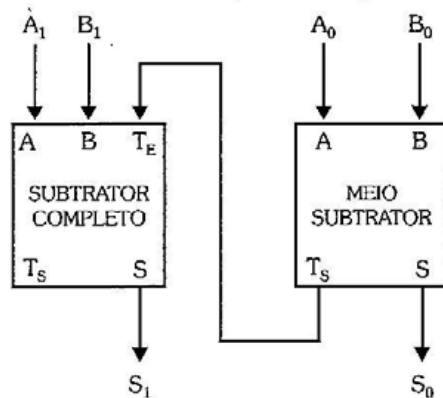
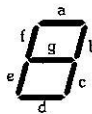



Figura 5.61

5.5 Quadro Resumo

Códigos					
Decimal	**BCD 8421**	**Excesso 3**	**Gray**	**2 entre 5**	**Jonhson**
0	0 0 0 0	0 0 1 1	0 0 0 0	0 0 0 1 1	0 0 0 0 0
1	0 0 0 1	0 1 0 0	0 0 0 1	0 0 1 0 1	0 0 0 0 1
2	0 0 1 0	0 1 0 1	0 0 1 1	0 0 1 1 0	0 0 0 1 1
3	0 0 1 1	0 1 1 0	0 0 1 0	0 1 0 0 1	0 0 1 1 1
4	0 1 0 0	0 1 1 1	0 1 1 0	0 1 0 1 0	0 1 1 1 1
5	0 1 0 1	1 0 0 0	0 1 1 1	0 1 1 0 0	1 1 1 1 1
6	0 1 1 0	1 0 0 1	0 1 0 1	1 0 0 0 1	1 1 1 1 0
7	0 1 1 1	1 0 1 0	0 1 0 0	1 0 0 1 0	1 1 1 0 0
8	1 0 0 0	1 0 1 1	1 1 0 0	1 0 1 0 0	1 1 0 0 0
9	1 0 0 1	1 1 0 0	1 1 0 1	1 1 0 0 0	1 0 0 0 0

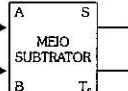
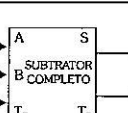
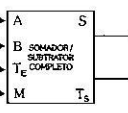
Display de 7 segmentos		
a, b, c, d, e, f, g	catodo comum	Cada segmento acende com 1 aplicado ao respectivo anodo.
	anodo comum	Cada segmento acende com 0 aplicado ao respectivo catodo.

Circuitos Aritméticos		
Meio Somador	A, B → MEIO SOMADOR → S, T_S	$S = A \oplus B$ $T_s = AB$
Meio Subtrator	A, B → MEIO SUBTRATOR → S, T_S	$S = A \oplus B$ $T_s = \overline{A}B$
Somador Completo	A, B, T_E → SOMADOR COMPLETO → S, T_S	$S = A \oplus B \oplus T_E$ $T_s = AB + (A \oplus B) \cdot T_E$ $T_s = AB + AT_E + BT_E$
Subtrator Completo	A, B, T_E → SUBTRATOR COMPLETO → S, T_S	$S = A \oplus B \oplus TE$ $T_s = \overline{A}B + \overline{(A \oplus B)} \cdot T_E$ $T_s = \overline{A}B + \overline{A}T_E + BT_E$
Somador/Subtrator Completo M = 0 → Somador M = 1 → Subtrator	A, B, T_E, M → SOMADOR/ SUBTRATOR COMPLETO → S, T_S	$S = A \oplus B \oplus TE$ $T_s = BT_E + (M \oplus A) \cdot (B + T_E)$

Tabela 5.23

5.6 Exercícios Propostos

5.6.1 - Elabore um Codificador Decimal/Binário para, a partir de um teclado com chaves numeradas de 0 a 3, fornecer nas saídas o código correspondente. Considere que as entradas das portas em vazio equivalem à aplicação de nível lógico 1.

5.6.2 - Projete um circuito combinacional para em um conjunto de 4 fios, fornecer nível 0 em apenas um deles por vez (estando os demais em nível 1), conforme seleção binária aplicada às entradas digitais.

5.6.3 - Elabore um decodificador 3 para 8 onde, conforme as combinações entre os 3 fios de entrada, 1 entre os 8 fios de saída é ativado (nível 1).

5.6.4 - Desenvolva um circuito que transforme do código BCD 8421 para o código de Johnson.

5.6.5 - Projete um decodificador do código Gray para o Excesso 3. Dê apenas as expressões simplificadas.

5.6.6 - Projete um decodificador para, a partir de um código binário, escrever a seqüência de 1 a 5 em um display de 7 segmentos catodo comum.

5.6.7 - Idem ao anterior, para escrever a seqüência da figura 5.62 em um display de 7 segmentos anodo comum.

CARACTERE	C	d	P	L	A	Y	E	r
CASO	0	1	2	3	4	5	6	7

Figura 5.62

5.6.8 - Monte a tabela e simplifique as expressões do decodificador do código Gray para hexadecimal, visualizado em um display de 7 segmentos catodo comum.

5.6.9 - Faça o projeto e desenhe o circuito para, a partir de um código binário, escrever a seqüência do sistema hexadecimal em um display de 7 segmentos anodo comum.

5.6.10 - Mostre como um bloco Somador Completo pode ser utilizado para efetuar a soma de 3 números de 1 bit.

5.6.11 - Esquematize, em blocos, um sistema subtrator para 2 números de 4 bits.

5.6.12 - Utilizando o sistema obtido no exercício 5.6.11, faça um estudo e conclua qual o resultado obtido no caso de o minuendo (A_3 A_2 A_1 A_0) ser menor que o subtraendo (B_3 B_2 B_1 B_0).

5.6.13 - Elabore um Meio Somador/ Meio Subtrator (M = 0 → Meio Somador e M = 1 → Meio Subtrator).

5.6.14 - Esquematize, em blocos, um sistema Somador/Subtrator Completo para 2 números de 4 bits.

5.6.15 - Estenda o circuito obtido no exercício resolvido nº 2 (item 5.48), para um de 4 bits.

5.6.16 - Utilizando blocos de Somadores Completos, elabore um sistema subtrator para 2 números de 2 bits.

5.6.17 - Utilizando blocos de Somadores Completos, elabore um sistema para 2 números de 2 bits que faça soma ou subtração, conforme o nível aplicado a uma entrada de controle M (M = 0 → soma e M = 1 → subtração).

CAPÍTULO 6

Flip-Flop. Registradores e Contadores

6.1 Introdução

O campo da Eletrônica Digital é basicamente dividido em duas áreas: lógica combinacional e lógica seqüencial.

Os circuitos combinacionais, como vimos até aqui, apresentam as saídas, única e exclusivamente, dependentes das variáveis de entrada.

Os circuitos seqüenciais têm as saídas dependentes das variáveis de entrada e/ou de seus estados anteriores que permanecem armazenados, sendo, geralmente, sistemas pulsados, ou seja, operam sob o comando de uma seqüência de pulsos denominada **clock**.

Neste capítulo, trataremos do estudo dos flip-flops e de circuitos nos quais fazem o papel de elemento principal.

6.2 Flip-Flops

De forma geral, podemos representar o flip-flop como um bloco onde temos 2 saídas: Q e $\overline{Q}$, entradas para as variáveis e uma entrada de controle (clock). A saída Q será a principal do bloco. A figura 6.1 ilustra um flip-flop genérico.

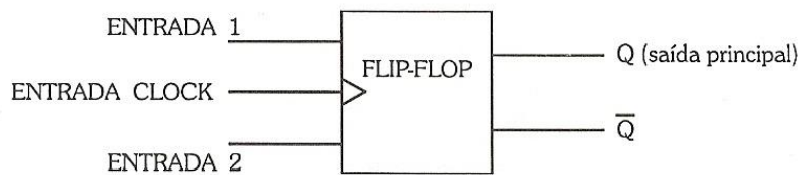



Figura 6.1

Este dispositivo possui basicamente dois estados de saída. Para o flip-flop assumir um destes estados é necessário que haja uma combinação das variáveis e do pulso de controle (clock). Após este pulso, o flip-flop permanecerá neste estado até a chegada de um novo pulso de clock e, então, de acordo com as variáveis de entrada, mudará ou não de estado.

Os dois estados possíveis são:

1) $Q=0 \rightarrow \overline{Q}=1$

2) $Q=1 \rightarrow \overline{Q}=0$

Vamos, a seguir, analisar alguns circuitos de flip-flops e suas respectivas operações.

6.2.1 Flip-Flop RS Básico

Primeiramente, vamos analisar o flip-flop RS básico, construído a partir de portas NE e inversores, cujo circuito é visto na figura 6.2.

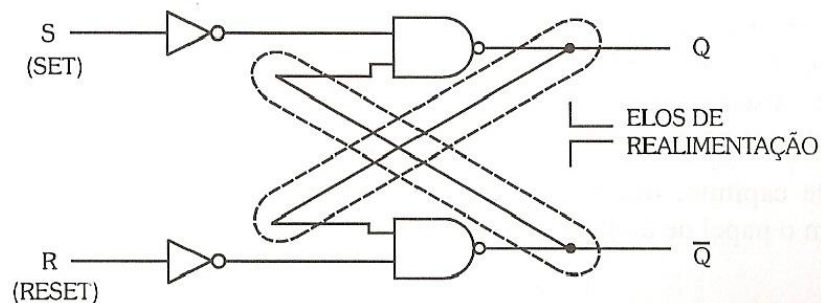



Figura 6.2

Notamos que estes elos de realimentação fazem com que as saídas sejam injetadas juntamente com as variáveis de entrada, ficando claro, então, que os estados que as saídas irão assumir dependerão de ambas.

Para analisarmos o comportamento do circuito, vamos construir a tabela da verdade, levando em consideração as 2 variáveis de entrada (S e R) e a saída Q anterior (Qa) à aplicação das entradas:

	S	R	Qa	Qf
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

Tabela 6.1

A saída que o flip-flop irá assumir (Qf), portanto, será em função das entradas S, R e da saída anterior (Qa).

Vamos, agora, analisar cada caso possível:

Caso 0: S = 0, R = 0 e Qa = 0 → $\overline{Q}a = 1$

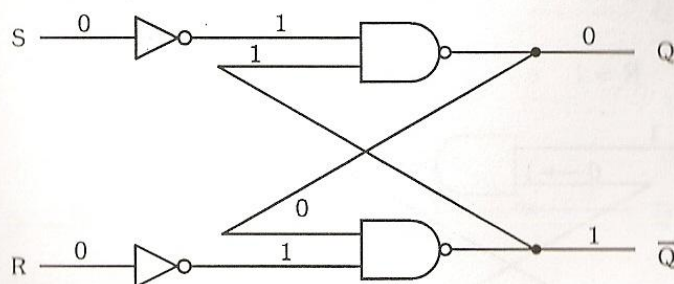



Figura 6.3

Podemos notar que este estado é estável, logo, o valor que a saída Q irá assumir será igual ao seu valor anterior à aplicação das entradas:

Qf = Qa = 0

Caso 1: S = 0, R = 0 e Qa = 1 → $\overline{Q}a = 0$

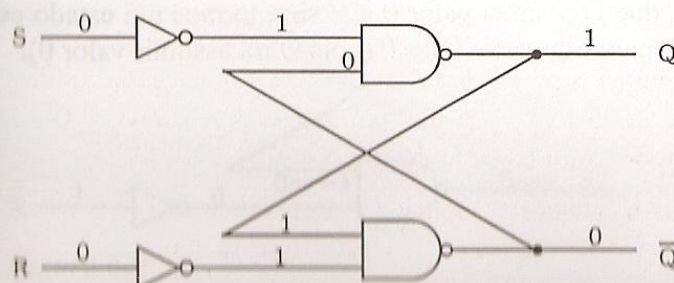



Figura 6.4

Este também será um estado estável, logo, o valor que a saída Q do flip-flop irá assumir será igual ao seu valor anterior:

Qf = Qa = 1

Caso 2: S = 0, R = 1 e $Qa = 0 \rightarrow \overline{Q}a = 1$

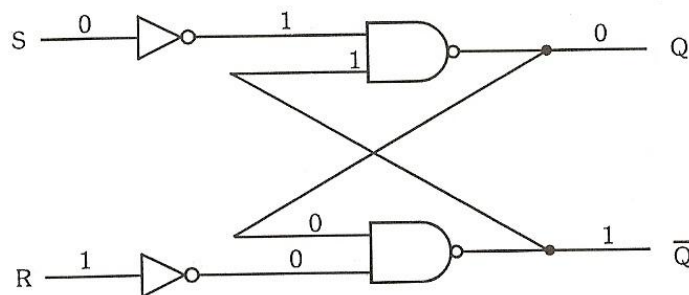



Figura 6.5

Este estado é estável, logo, Q irá assumir o valor 0:

Qf = 0.

Caso 3: S = 0, R = 1 e $Qa = 1 \rightarrow \overline{Q}a = 0$

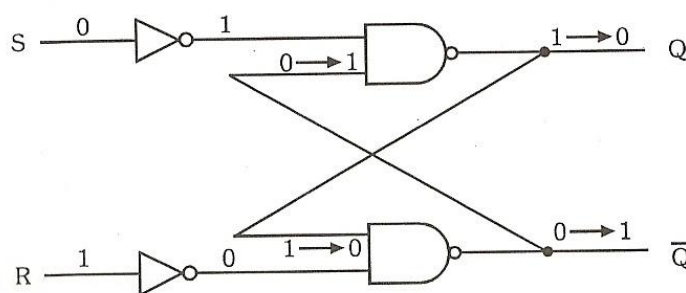



Figura 6.6

Notamos, agora, que a saída Q está num estado instável, pois $\overline{Q}$ irá mudar para 1, forçando assim que Q assuma valor 0 e aí sim, termos um estado estável, logo, podemos escrever para este caso: Qf = 0 (pois Q irá assumir valor 0).

Caso 4: S = 1,   R = 0 e $Qa = 0 \rightarrow \overline{Q}a = 1$

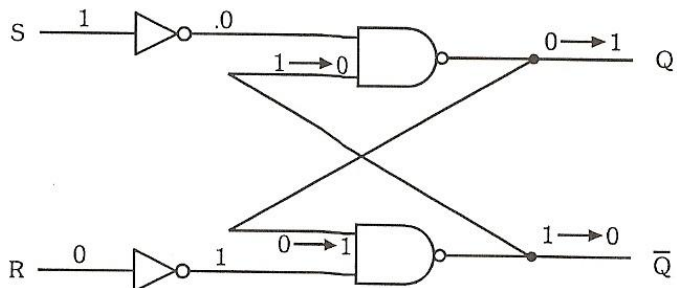



Figura 6.7

Notamos que este é um estado instável, pois Q irá assumir forçosamente valor 1 e, por conseguinte, $\overline{Q}$ assumirá valor 0, logo, podemos escrever: Qf = 1.

Caso 5: S = 1,   R = 0 e $Qa = 1 \rightarrow \overline{Q}a = 0$

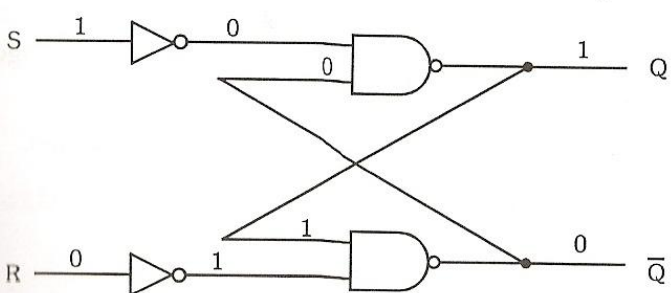



Figura 6.8

Notamos que este é um estado estável, logo, podemos escrever para este caso: Qf = 1.

Caso 6: S = 1,   R = 1 e $Qa = 0 \rightarrow \overline{Q}a = 1$

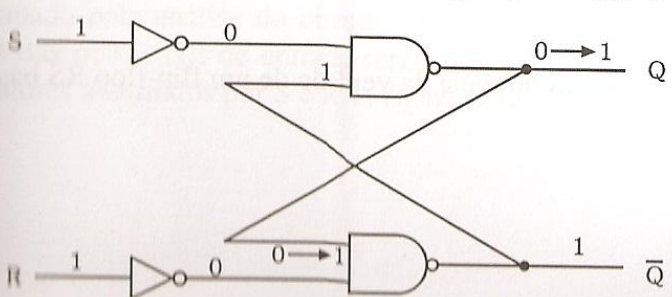



Figura 6.9

Notamos que este é um estado instável, pois Q forçosamente irá assumir valor 1. Notamos, também, que $\overline{Q}$ irá assumir valor 1. Podemos escrever para este caso: $Qf = \overline{Q}f = 1$.

Este caso não poderá ser permitido na entrada, pois forçará o flip-flop a assumir um estado de saída, no qual a saída Q será igual à saída complementar $\overline{Q}$.

Caso 7: S = 1, R = 1 e $Qa = 1 \rightarrow \overline{Q}a = 0$

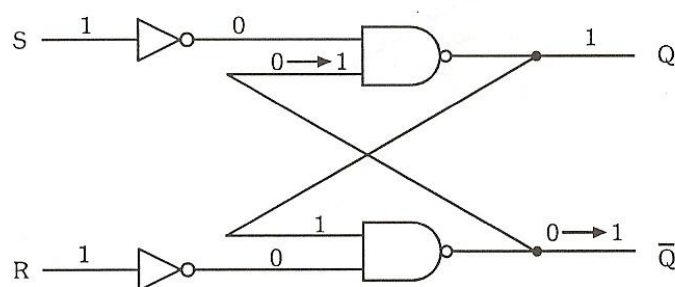



Figura 6.10

Notamos que esta é uma situação instável e análoga ao caso 6, logo, esta também será uma situação não permitida.

A tabela 6.2 apresenta o resultado da análise de todos estes casos.

S	R	Qa	Qf	$\overline{Q}f$	
0	0	0	0	1	} fixa Qf = Qa
0	0	1	1	0	
0	1	0	0	1	} fixa Qf em 0
0	1	1	0	1	
1	0	0	1	0	} fixa Qf em 1
1	0	1	1	0	
1	1	0	1	1	} não permitido
1	1	1	1	1	

Tabela 6.2

Podemos, então, resumir a tabela da verdade de um flip-flop RS básico:

A entrada S é denominada **Set**, pois quando acionada (nível 1), passa a saída para 1 (estabelece ou fixa 1), e a entrada R é denominada **Reset**, pois quando acionada (nível 1), passa a saída para 0 (recompõe ou zera o flip-flop). Estes termos são muito usuais na área de eletrônica digital, sendo provenientes do idioma inglês.

Este circuito irá mudar de estado apenas no instante em que mudam as variáveis de entrada. Veremos em seguida, como é o circuito de um flip-flop RS que tem sua mudança de estado controlada pela entrada de clock.

6.2.2 Flip-Flop RS com Entrada Clock

Para que o flip-flop RS básico seja controlado por uma seqüência de pulsos de clock, basta trocarmos os 2 inversores por portas NE, e às outras entradas destas portas, injetarmos o clock. O circuito, com estas modificações, é visto na figura 6.11.

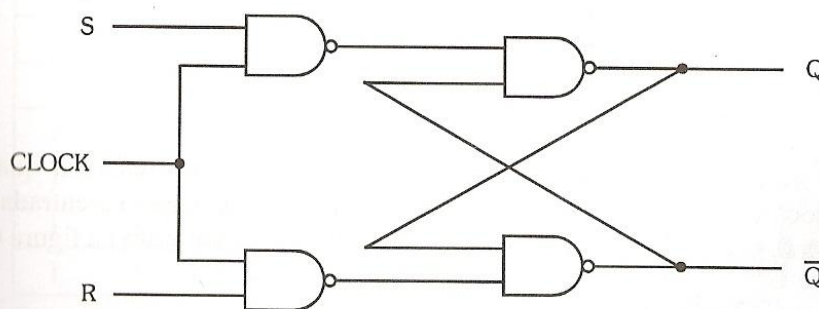



Figura 6.11

Neste circuito, quando a entrada do clock for igual a 0, o flip-flop irá permanecer no seu estado, mesmo que variem as entradas S e R. Isso pode ser confirmado pela análise do circuito, onde concluímos que para clock = 0, as saídas das portas NE de entrada serão sempre iguais a 1, independentemente dos valores assumidos por S e R. A figura 6.12 ilustra esta situação.

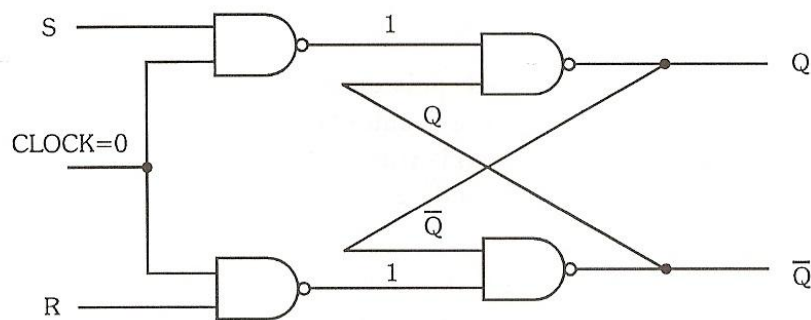



Figura 6.12

Quando a entrada clock assumir valor 1, o circuito irá comportar-se como um flip-flop RS básico, pois as portas NE de entrada funcionarão como os inversores do circuito anteriormente visto. A tabela 6.4 resume a operação deste flip-flop em função da entrada clock.

CK	Qf
0	Qa
1	RS básico

Tabela 6.4

De maneira geral, podemos concluir que o circuito irá funcionar quando a entrada clock assumir valor 1 e manterá travada esta saída quando a entrada clock passar para 0. O flip-flop RS pode ser representado pelo bloco visto na figura 6.13.

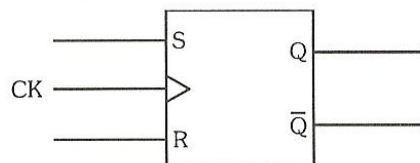



Figura 6.13

6.2.3 Flip-Flop JK

O flip-flop JK nada mais é que um flip-flop RS realimentado da maneira mostrada na figura 6.14.

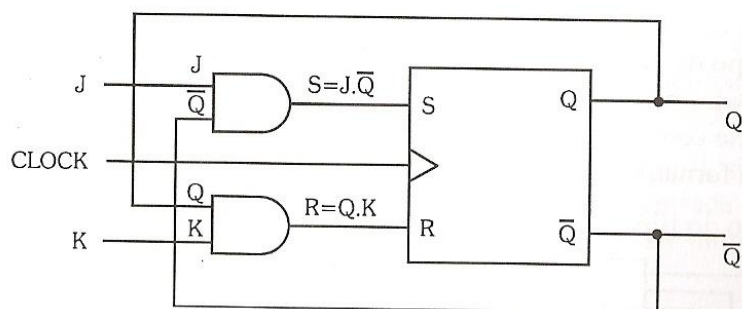



Figura 6.14

Vamos, agora, levantar a tabela da verdade do flip-flop JK com entrada clock igual a 1:

J	K	Qa	$\overline{Qa}$	S	R	Qf	
0	0	0	1	0	0	Qa	} Qa
0	0	1	0	0	0	Qa	
0	1	0	1	0	0	Qa (Qa = 0)	} 0
0	1	1	0	0	1	0	
1	0	0	1	1	0	1	} 1
1	0	1	0	0	0	Qa(Qa = 1)	
1	1	0	1	1	0	$\overline{Qa}$(Qa = 0)	} $\overline{Qa}$
1	1	1	0	0	1	$\overline{Qa}$(Qa = 1)	

Tabela 6.5

A tabela simplificada resultante será:

J	K	Qf
0	0	Qa
0	1	0
1	0	1
1	1	$\overline{Qa}$

Tabela 6.6

No caso J = 1 e K = 1, para obter-se Qf = $\overline{Qa}$ é necessário que a entrada clock volte à situação 0 em um tempo conveniente após a aplicação das

entradas, pois, caso contrário, a saída entrará em constante mudança (oscilação), provocando novamente uma indeterminação. Este tempo deve levar em conta o **tempo de atraso de propagação** de cada porta lógica, a ser abordado no capítulo 9. Outra possibilidade, para melhor desempenho, é a de inserir blocos de atraso em série com as linhas de realimentação no circuito e comutar a entrada clock da mesma forma, ou seja, para se obter na saída $Qf = \overline{Q}a$.

O circuito do flip-flop JK é constituído da seguinte forma:

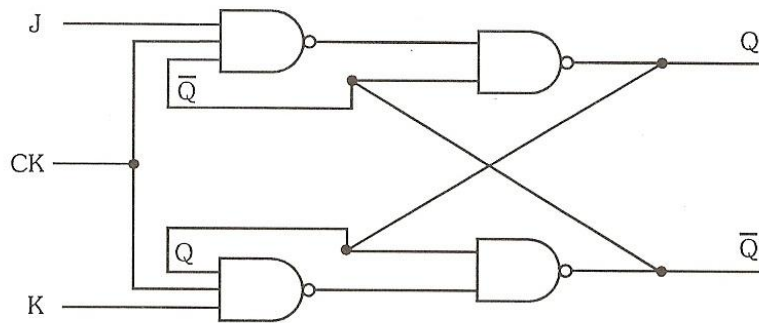



Figura 6.15

6.2.4 Flip-Flop JK com Entradas Preset e Clear

O flip-flop JK poderá assumir valores Q = 1 ou Q = 0 mediante a utilização das entradas **Preset (PR)** e **Clear (CLR)**. Estas entradas são inseridas no circuito, conforme mostra a figura 6.16.

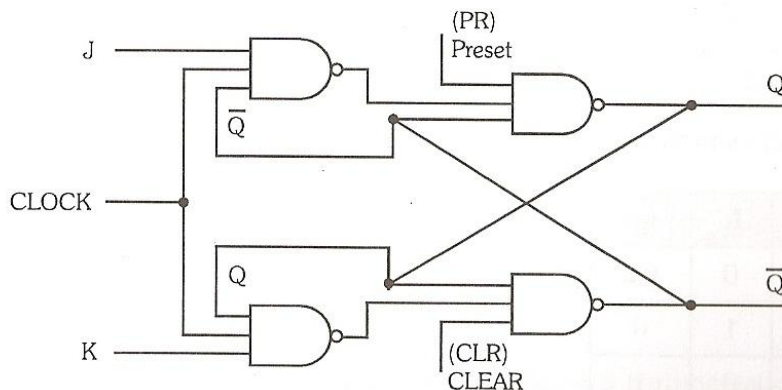



Figura 6.16

Analisando este circuito, podemos notar que com a entrada clock igual a 0 e conseqüente bloqueio da passagem das entradas J e K, podemos impor ao circuito saída Q igual a 1 através da aplicação à entrada Preset de nível 0. De

forma análoga, podemos fazer Q = 0 mediante aplicação à entrada Clear de nível 0. Podemos notar também que com essas entradas permanecendo iguais a 1, o circuito funciona normalmente como sendo um flip-flop JK.

As entradas Preset e Clear não podem assumir valor 0, simultaneamente, pois acarretaria à saída uma situação não permitida. A entrada Clear é também denominada Reset, termo este, da mesma forma que os outros, derivado do inglês.

A tabela 6.7 resume a atuação das entradas Preset e Clear para esse flip-flop.

CLR	PR	Qf
0	0	não permitido
0	1	0
1	0	1
1	1	funcionamento normal

Tabela 6.7

Podemos, para facilitar, utilizar um bloco representativo como o mostrado na figura 6.17, com todas as entradas.

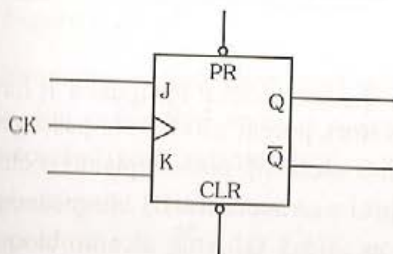



Figura 6.17

Os círculos na simbologia do bloco, indicam que as entradas Preset e Clear são ativas em 0, ou seja, funcionam respectivamente com nível 0 aplicado. Para utilizar estas entradas como ativas em 1, basta colocar inversores e na simbologia excluir os círculos aqui empregados.

6.2.5 Flip-Flop JK Mestre-Escravo

O flip-flop JK apresenta uma característica indesejável. Quando o clock for igual a 1, teremos o circuito funcionando como sendo um circuito combinacional, pois haverá a passagem das entradas J, K e também da realimentação. Nessa situação, se houver uma mudança nas entradas J e K, o circuito apresentará uma nova saída, podendo alterar seu estado tantas vezes quantas alterarem os estados das entradas J e K.

Para resolver esse problema, foi criado o flip-flop **JK Mestre-Escravo (JK Master-Slave)** cujo circuito é apresentado na figura 6.18.

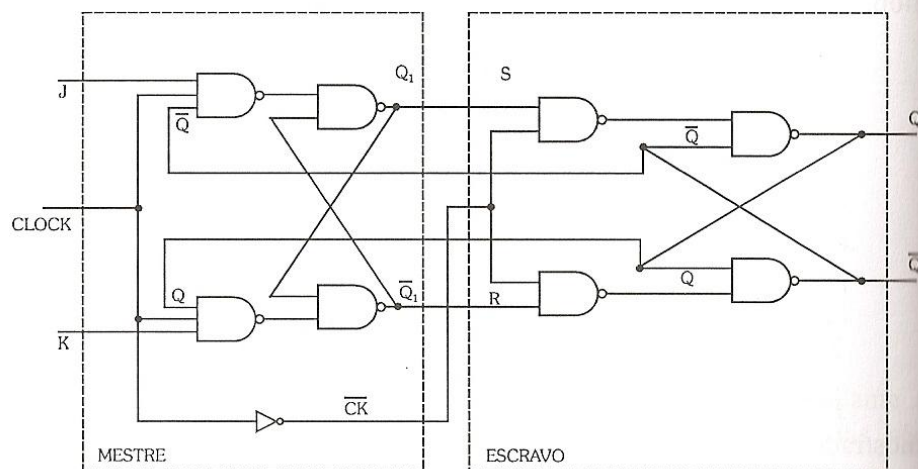



Figura 6.18

Primeiramente, devemos notar que quando o clock for igual a 1, haverá a passagem das entradas J e K (circuito mestre), porém não haverá passagem das saídas Q_1 e $\overline{Q}_1$ (entradas S e R do circuito escravo), pois enquanto o clock do circuito mestre for igual a 1, no circuito escravo será 0, bloqueando suas entradas. Quando o clock passar para 0, as saídas Q_1 e $\overline{Q}_1$ ficarão bloqueadas no último estado assumido e entrarão em R e S desbloqueadas, mudando o estado do circuito escravo e conseqüentemente das saídas Q e $\overline{Q}$. Nota-se aqui, que o problema da variação das entradas J e K foi resolvido, pois o circuito só reconhecerá as entradas J e K no instante da passagem do clock para 0.

O gráfico da figura 6.19 exemplifica uma das possíveis mudanças de estado do flip-flop JK Mestre-Escravo. Verificamos a atuação do clock e todo o processo de mudança de estados.

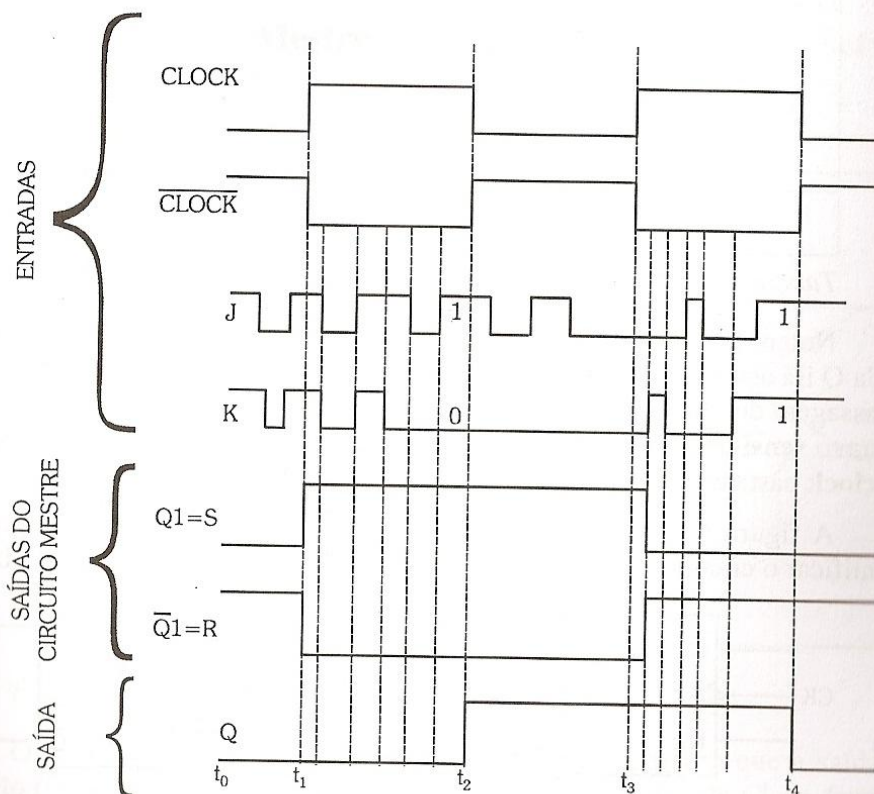



Figura 6.19

Enquanto o clock permanece em 0, notamos que J e K podem variar à ontade que o flip-flop manterá a saída constante, pois Q_1 e $\overline{Q}_1$ (S e R) ermanecerão fixos (instantes de t_0 a t_1 e de t_2 a t_3). No momento em que o ock passa para 1 (instantes t_1 e t_3), os pontos Q_1 e $\overline{Q}_1$ irão mudar de tado conforme as entradas J e K, porém a saída Q permanecerá constante, is a entrada de clock do circuito escravo ($\overline{CK}$) estará em 0 (instantes de t_1 a e de t_3 a t_4). O circuito mestre irá assumir o estado que for imposto pelas tradas J e K no momento em que o clock mudar para 0 (t_2), permanecendo ste estado até que o clock volte a mudar (t_3). A saída assumida pelo circuito estre irá impor ao circuito escravo o seu estado, e este só irá mudar na óxima vez em que o clock mudar de 1 para 0 (t_4).

A tabela 6.8 resume a operação do flip-flop JK Mestre-Escravo:

J	K	Qf
0	0	Qa
0	1	0
1	0	1
1	1	$\overline{Qa}$

Tabela 6.8

Notamos que esta tabela é idêntica à de um flip-flop JK básico, porém a saída Q irá assumir valores, conforme a situação das entradas JK, somente após a passagem do clock para 0. Assim sendo, o circuito é denominado JK Mestre-Escravo **sensível à descida de clock**. Para obter um circuito **sensível à subida de clock** basta colocarmos um inversor interno à entrada clock.

A figura 6.20 mostra o bloco JK Mestre-Escravo e a simbologia para identificar o circuito sensível à descida de clock (a) e à subida de clock (b).

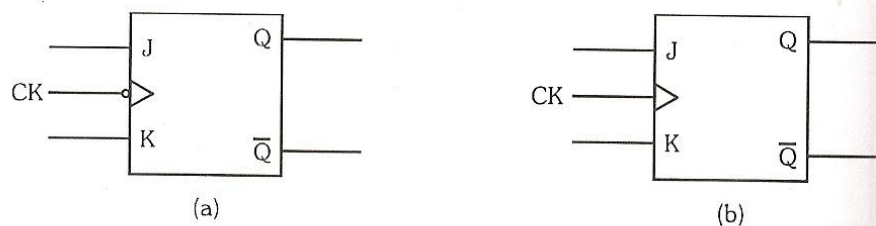



Figura 6.20

O círculo no bloco da figura 6.20(a), indica que o clock é ativo quando passa de 1 para 0.

6.2.6 Flip-Flop JK Mestre-Escravo com Entrada Preset e Clear

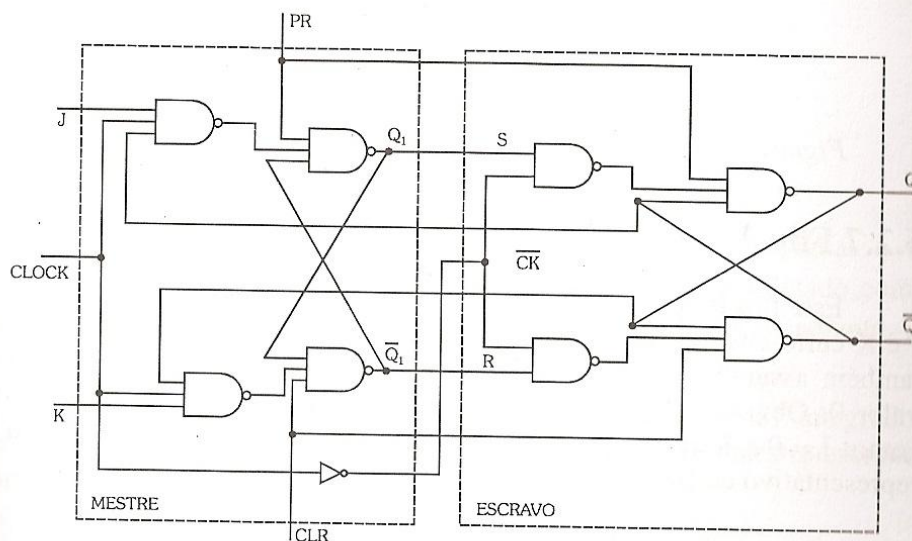



Figura 6.21

O controle de Preset, quando assumir valor 0, fará com que a saída do circuito (Q) assuma valor 1. O mesmo ocorre com o controle de Clear, fazendo com que a saída assuma valor 0. Notamos que ambos, por estarem ligados simultaneamente aos circuitos Mestre e Escravo, atuam independentemente da entrada clock. Todas as situações possíveis são vistas na tabela 6.9.

CLR	PR	Qf
0	0	não permitido
0	1	0
1	0	1
1	1	funcionamento normal

Tabela 6.9

A figura 6.22 mostra o bloco representativo do flip-flop JK Mestre-Escravo com as entradas Preset e Clear ativas em 0.

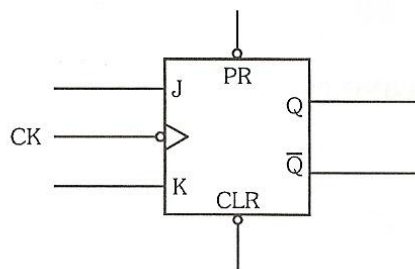



Figura 6.22

6.2.7 Flip-Flop Tipo T

Este flip-flop é obtido a partir de um JK Mestre-Escravo com as entradas J e K curto-circuitadas (uma ligada à outra), logo quando J assumir valor 1, K também assumirá valor 1, e quando J assumir valor 0, K também assumirá valor 0. Obviamente, no caso desta ligação, não irão ocorrer nunca entradas como: J = 0 e K = 1; J = 1 e K = 0. A figura 6.23 mostra a ligação e o bloco representativo do flip-flop tipo T obtido.

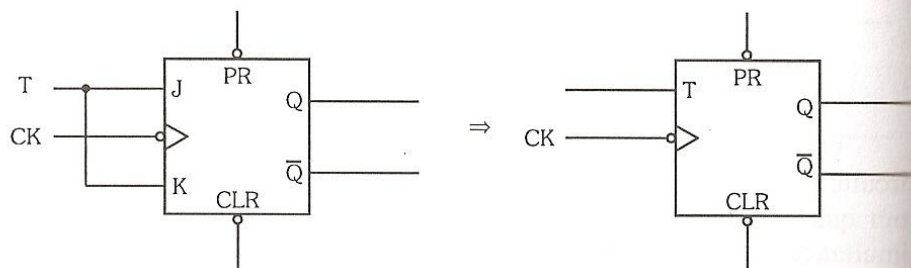



Figura 6.23

A tabela da verdade completa será, então:

J	K	T	Qf
0	0	0	Qa
0	1	não existe	/
1	0	não existe	/
1	1	1	$\overline{Qa}$

Tabela 6.10

Eliminando os casos não existentes, obtemos a tabela da verdade do flip-flop do tipo T:

T	Qf
0	Qa
1	$\overline{Qa}$

Tabela 6.11

Devido ao fato de o flip-flop tipo T, com a entrada T igual a 1, complementar a saída ($\overline{Qa}$) a cada descida de clock, este será utilizado como célula principal dos **contadores assíncronos** que serão estudados adiante. A sigla **T** vem de **Toggle**. (comutado).

O flip-flop tipo T, não é encontrado na série de circuitos integrados comerciais, sendo na prática montado à partir de um JK mestre-escravo, conforme já visto.

6.2.8 Flip-Flop Tipo D

É obtido a partir de um flip-flop JK Mestre-Escravo com a entrada K invertida (por inversor) em relação a J. Logo, neste flip-flop, teremos as seguintes entradas possíveis: J = 0 e K = 1; J = 1 e K = 0. Obviamente, não irão ocorrer os casos: J = 0 e K = 0; J = 1 e K = 1. A figura 6.24 mostra como este é obtido e seu bloco representativo.

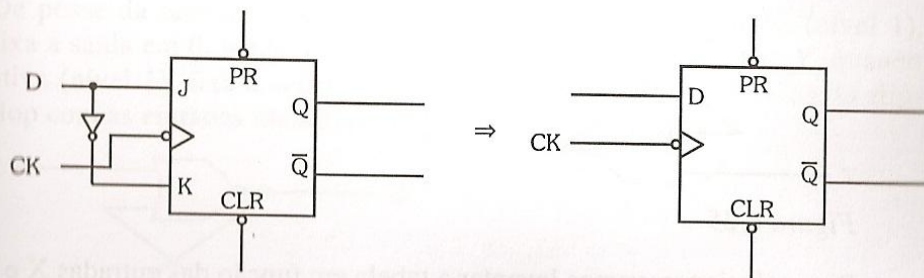



Figura 6.24

A tabela da verdade completa será, então:

J	K	D	Qf
0	0	não existe	/
0	1	0	0
1	0	1	1
1	1	não existe	/

Tabela 6.12

Eliminando os casos não existentes, obtemos a tabela do flip-flop tipo D.

D	Qf
0	0
1	1

Tabela 6.13

Pela capacidade de passar para a saída (Qf) e armazenar o dado aplicado na entrada D, este flip-flop será empregado como célula de **registradores de deslocamento** e em outros sistemas de memória, a serem estudados adiante. A sigla **D** vem de **Data** (dado), termo original em inglês.

6.2.9 Exercícios Resolvidos

1 - Levante a tabela da verdade do flip-flop da figura 6.25 e identifique as entradas S e R.

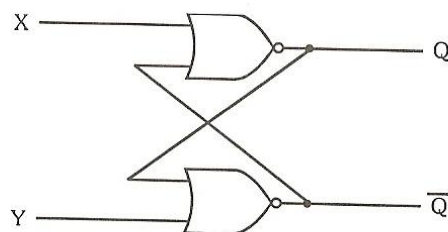



Figura 6.25

Para solucionar, vamos levantar a tabela em função das entradas X e Y:

1º) X = 0 e Y = 0: Para Qa = 0 ($\overline{Qa}$ = 1), a saída não se alterará (Qf = 0), pois os níveis são concordantes com a operação das portas NOU envolvidas, sendo uma situação estável. O mesmo ocorrerá com Qa = 1 ($\overline{Qa}$ = 0), onde Qf = 1.

2º) X = 0 e Y = 1: O nível 1 aplicado em Y fará com que a respectiva porta NOU apresente saída igual a 0 ($\overline{Qf} = 0$) e conseqüentemente Qf = 1, pois o nível 0 de $\overline{Qf}$ será aplicado juntamente ao nível 0 da entrada X, na outra porta NOU, forçando a saída Qf = 1, independentemente dos estados de saída anteriores.

3º) X = 1 e Y = 0: Este caso será exatamente oposto ao anterior, pois a aplicação do nível 1 em X fará com que a saída assuma valor 0 (Qf = 0), independentemente das situações das saídas antes da aplicação das entradas.

4º) X = 1 e Y = 1: Estes níveis aplicados forçarão ambas as saídas a assumir valor 0 (Qf = $\overline{Qf}$ = 0), caracterizando-se num caso não permitido.

A tabela 6.14 apresenta todos os resultados, conforme a análise efetuada.

X	Y	Qf
0	0	Qa
0	1	1
1	0	0
1	1	✕

Tabela 6.14

De posse da tabela, concluímos que a entrada X quando ativa (nível 1), fixa a saída em 0, tendo a função de Reset (X = R), e a entrada Y, quando ativa (nível 1), fixa a saída em 1, tendo a função de Set (Y = S). O flip-flop com as entradas identificadas é visto na figura 6.26.

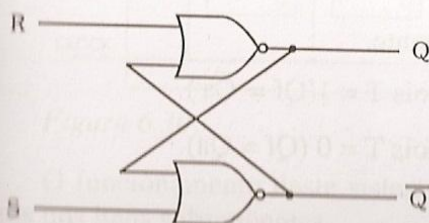



Figura 6.26

Determine as formas de onda das saídas Q e $\overline{Q}$ do flip-flop tipo T visto na figura 6.27, em função dos sinais aplicados nas entradas.

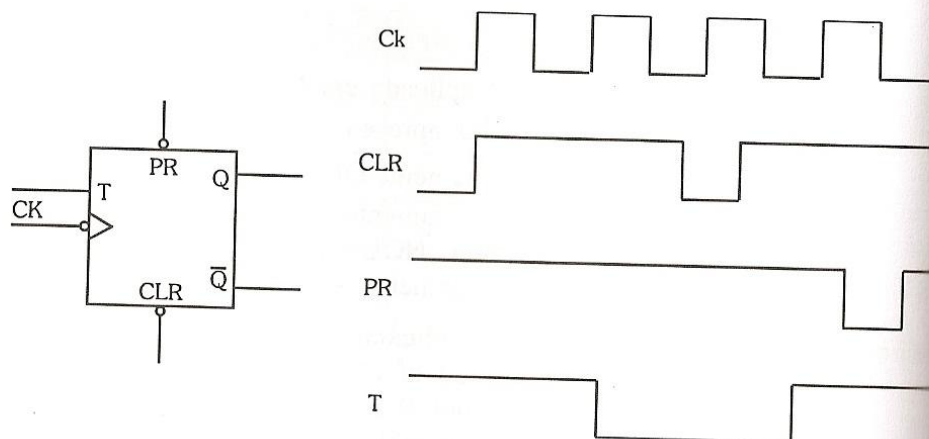



Figura 6.27

Para determinarmos estas formas de onda, devemos considerar a atuação das entradas Preset (PR = 0 → Qf = 1), Clear (CLR = 0 → Qf = 0) e do sinal T na descida de clock (T = 0 → Qf = Qa e T = 1 → Qf = $\overline{Qa}$). Assim sendo, a figura 6.28 apresenta os sinais Q e $\overline{Q}$, com o casos de mudança indicados.

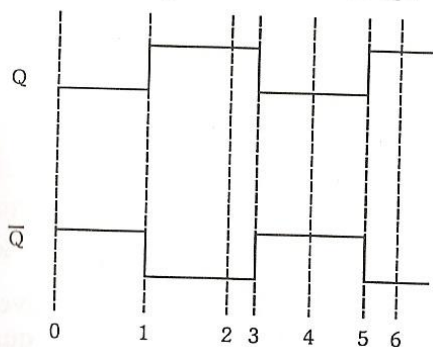



Figura 6.28

Análise dos casos de mudança:

0 - Qf = 0, pois CLR = 0 inicialmente.

1 - 1ª descida de clock: Qf = 1, pois T = 1(Qf = $\overline{Qa}$).

2 - 2ª descida de clock: Qf = 1, pois T = 0 (Qf = Qa).

3 - Qf = 0, pois CLR = 0.

4 - 3ª descida de clock: Qf = 0, pois T = 0.

5 - Qf = 1, pois PR = 0.

6 - 4ª descida de clock: Qf = 1, pois PR = 0.

6.3 Registradores de Deslocamento

Como vimos, o flip-flop pode armazenar durante o período em que sua entrada clock for igual a 0, um bit apenas (saída Q). Porém, se necessitarmos guardar uma informação de mais de um bit, o flip-flop irá tornar-se insuficiente. Para isso utilizamo-nos de um sistema denominado **Registrador de Deslocamento (Shift Register).** Trata-se de um certo número de flip-flops tipo JK mestre-escravo ligado de tal forma que as saídas de cada bloco sejam aplicadas nas entradas J e K respectivas do flip-flop seguinte, sendo o primeiro, com suas entradas ligadas na forma de um flip-flop tipo D. A figura 6.29 representa um Registrador de Deslocamento generalizado para N + 1 bits.

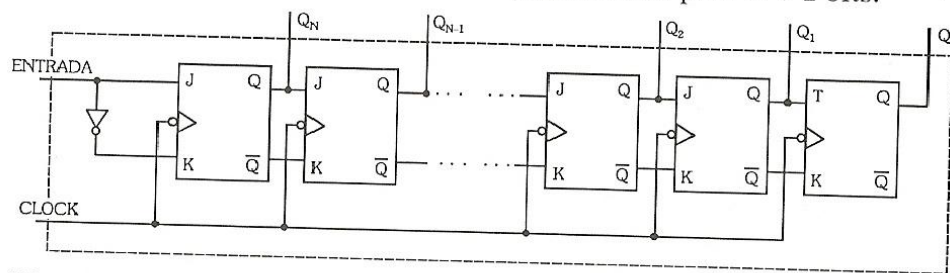



Figura 6.29

Pelo fato de os flip-flops envolvidos atuarem como os do tipo D, este circuito, para facilitar, pode ser construído apenas com flip-flops do tipo D. A figura 6.30 mostra a mesma estrutura geral, porém composta apenas com flip-flops tipo D.

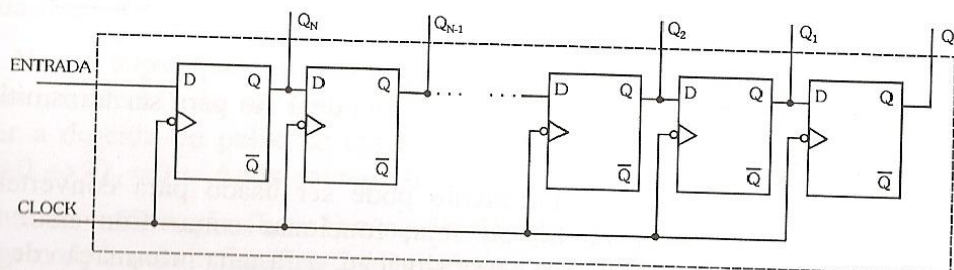



Figura 6.30

O funcionamento deste sistema, juntamente com suas aplicações, serão vistos nos itens subseqüentes.

6.3.1 Conversor Série-Paralelo

Antes de estudarmos o comportamento do Registrador de Deslocamento como **Conversor Série-Paralelo**, vamos explicar o que significa **informação série** e **informação paralela.**

Chamamos de informação paralela a uma informação na qual todos os bits se apresentam simultaneamente. Uma informação paralela necessita tantos fios quantos forem os bits contidos nela, além, logicamente, do fio referencial do sistema (terra). Para exemplificar, vamos utilizar uma informação de 4 bits:

I_3	I_2	I_1	I_0
1	0	1	0

Figura 6.31

Notamos que esta informação necessita de 4 fios para ser transmitida ou inserida no bloco.

Informação série é aquela que utiliza apenas 1 fio, sendo que os bits de informação vêm seqüencialmente, um após o outro. Como exemplo, vamos utilizar a mesma informação, porém em série:

I_3	I_2	I_1	I_0
1	0	1	0

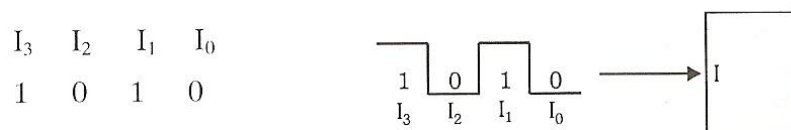



Figura 6.32

Notamos que esta informação necessita de 1 fio para ser transmitida ou inserida no bloco.

O Registrador de Deslocamento pode ser usado para converter uma informação série em paralela, ou seja, funcionar como Conversor Série-Paralelo. A configuração básica nessa situação, para uma informação de 4 bits, é vista na figura 6.33.

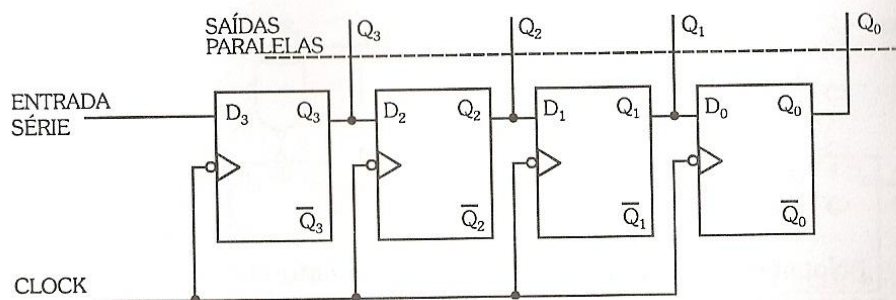



Figura 6.33

Como exemplo, vamos aplicar a informação série I = 1010 ($I_3 I_2\ I_1\ I_0$) à entrada série do registrador e analisar as saídas Q_0, Q_1, Q_2 e Q_3, após os pulsos de clock. Deve-se ressaltar que estes flip-flops atuam como mestre-escravo e têm sua comutação no instante da descida do pulso de clock. Assim sendo, temos:

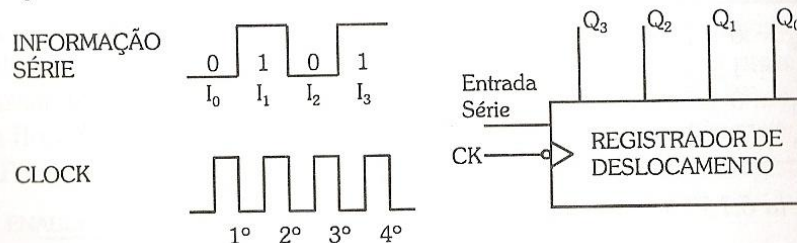



Figura 6.34

Entraremos com a informação (1010), como mostrado na figura 6.34 na entrada série, e os pulsos de clock na respectiva entrada (CK).

Vamos supor que, inicialmente, as saídas Q_3 , Q_2 , Q_1 e Q_0 do registrador estejam em nível 0. Ao ser injetado na entrada o 1º bit de informação ($I_0 = 0$) e houver a descida do pulso de clock, o flip-flop 3 irá apresentar na saída 0 ($D_3 = 0 \rightarrow Q_3 = 0$). Após este pulso de clock, aparecerá na entrada, o bit seguinte de informação ($I_1 = 1$) e na descida do 2º pulso de clock, teremos a passagem de I_0 para o flip-flop 2 ($D_2 = 0 \rightarrow Q_2 = 0$) e Q_3 assumirá o valor do bit de informação I_1 (entrada série = $D_3 = 1 \rightarrow Q_3 = 1$).

Após a descida do 3º pulso de clock, ficaremos com a seguinte situação:

⇨ $Q_1 = 0 \quad (D_1 = Q_2 = 0 \rightarrow Q_1 = 0)$,

⇨ $Q_2 = 1 \quad (D_2 = Q_3 = 1 \rightarrow Q_2 = 1)$ e

⇨ $Q_3 = 0 \quad (D_3 = I_2 = 0 \rightarrow Q_3 = 0)$.

Após o 4º pulso de clock, teremos a seguinte situação:

⇨ $Q_0 = 0 \quad (D_0 = Q_1 = 0 \rightarrow Q_0 = 0)$

⇨ $Q_1 = 1 \quad (D_1 = Q_2 = 1 \rightarrow Q_1 = 1)$

⇨ $Q_2 = 0 \quad (D_2 = Q_3 = 0 \rightarrow Q_2 = 0)$

⇨ $Q_3 = 1 \quad (D_3 = I_3 = 1 \rightarrow Q_3 = 1)$

Notamos que, após o 4º pulso de clock, a informação I estará armazenada no registrador de deslocamento e aparecerá nas saídas Q_3, Q_2, Q_1 e Q_0 como sendo uma informação paralela.

Para resumir, vamos representar toda a seqüência sob a forma da tabela da verdade:

Informação	descidas de clock	Q_3	Q_2	Q_1	Q_0
$I_0 = 0$	1ª	0	0	0	0
$I_1 = 1$	2ª	1	0	0	0
$I_2 = 0$	3ª	0	1	0	0
$I_3 = 1$	4ª	1	0	1	0

Tabela 6.15

É pelo motivo de deslocar a informação a cada pulso de clock que esse dispositivo é denominado Registrador de Deslocamento.

6.3.2 Conversor Paralelo-Série

Para entrarmos com uma informação paralela, necessitamos de um registrador que apresente entradas Preset e Clear, pois é através destas que fazemos com que o Registrador armazene a informação paralela. O registrador com estas entradas é visto na figura 6.35.

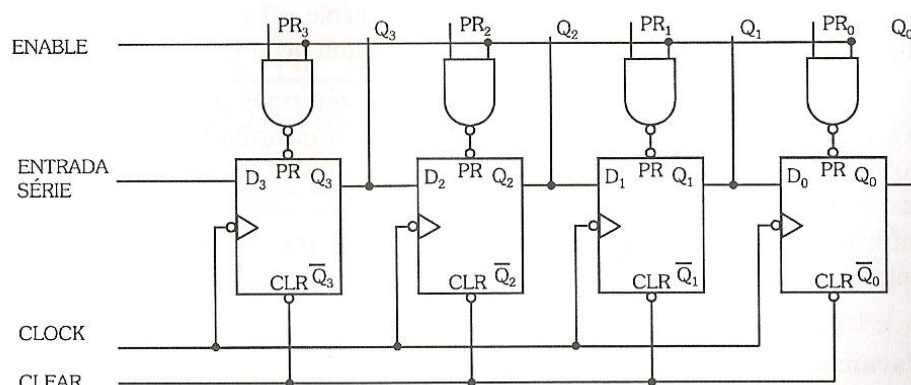



Figura 6.35

Primeiramente, vamos estudar o funcionamento da entrada **ENABLE**. Quando a entrada enable estiver em 0, as entradas preset (PR) dos flip-flops assumirão, respectivamente, níveis 1, fazendo com que o registrador atue normalmente. Quando a entrada enable for igual a 1, as entradas preset dos flip-flops assumirão os valores complementares das entradas PR_3, PR_2, PR_1 e PR_0, logo, os flip-flops irão assumir os valores que estiverem, respectivamente, em PR_3, PR_2, PR_1 e PR_0. Para entendermos melhor, vamos analisar uma célula do registrador:

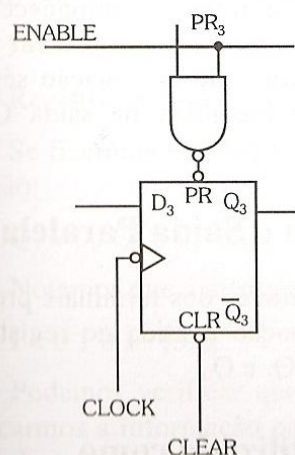



Figura 6.36

Para zerar (clear) o flip-flop ($Q_3 = 0$), vamos inicialmente, aplicar nível 0 à entrada clear.

Com enable = 0, a entrada PR do flip-flop irá assumir nível 1 e este irá ter um funcionamento normal como célula do registrador de deslocamento em questão, mantendo a saída no estado em que se encontra.

Com enable = 1 e PR3 = 0, a entrada PR do flip-flop assumirá nível 1, logo, a saída Q_3 manterá o seu estado ($Q_3 = 0$). Com enable = 1 e $PR_3 = 1$, a entrada PR do flip-flop assumirá nível 0, forçando a saída a assumir nível 1 ($Q_3 = 1$).

Após essa análise, concluímos que, se zerarmos o registrador (aplicando 0 à entrada clear), e logo após introduzirmos a informação paralela (I_3, I_2, I_1 e I_0) pelas entradas PR_3, PR_2, PR_1 e PR_0, as saídas Q_3, Q_2, Q_1 e Q_0 assumirão respectivamente os valores da informação. Essa maneira de entrarmos com a informação no registrador é chamada entrada paralela de informação, sendo a entrada enable responsável pela habilitação da mesma.

Para que o registrador de deslocamento funcione como **Conversor Paralelo-Série**, necessitamos zerá-lo e em seguida, introduzir a informação como já descrito, recolhendo na saída Q_0 a mesma informação de modo série. É fácil de notar que a saída Q_0 assume primeiramente o valor I_0 e a cada descida do pulso de clock, irá assumir seqüencialmente os valores I_1, I_2 e I_3.

6.3.3 Registrador de Entrada Série e Saída Série

Podemos utilizar o registrador de deslocamento com entrada série e o conseqüente armazenamento da informação no mesmo, e recolher a informação também de modo série. Notamos que nessa aplicação, após a entrada da informação, se inibirmos a entrada de clock, a informação permanecerá no registrador até que haja uma nova entrada. Assim sendo, é fácil observar que o registrador funcionou como uma memória. A entrada de informação série se faz na entrada série do registrador e pode ser recolhida na saída Q_0 do registrador.

6.3.4 Registrador de Entrada Paralela e Saída Paralela

A entrada paralela, como já visto, se faz através dos terminais preset e clear. Se inibirmos a entrada de clock, a informação contida no registrador pode ser acessada pelos terminais de saída Q_3, Q_2, Q_1 e Q_0.

6.3.5 Registrador de Deslocamento Utilizado como Multiplicador ou Divisor por 2

Como vimos, se entrarmos com uma informação num registrador de deslocamento, teremos as seguintes situações nas saídas:

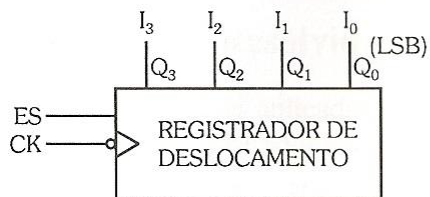



Figura 6.37

Se essa informação for considerada um número binário e deslocarmos o registrador uma casa à direita, entrando com 0 na entrada série, teremos a seguinte situação:

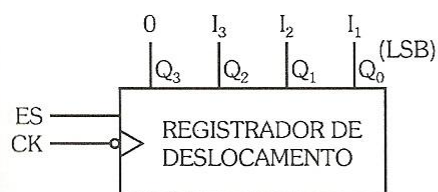



Figura 6.38

Podemos notar que essa operação, em binário, significa dividirmos um número por 2. Para exemplificar, vamos utilizar a informação:

$I = 1\ 0\ 1\ 0\ (10_{10})$

Registrador ⇨ $Q_3 = 1,\ \ Q_2 = 0,\ \ Q_1 = 1$ e $Q_0 = 0$

Se fizermos um deslocamento para a direita, teremos na saída a seguinte situação:

$Q_3 = 0,\ Q_2 = 1,\ Q_1 = 0,\ Q_0 = 1$

Notamos que a informação recolhida na saída será:

$I = 0\ 1\ 0\ 1\ \ (5_{10})$

Podemos verificar que o número foi dividido por 2. Esta operação de deslocarmos a informação para a direita é também conhecida por **Shift Right**, termo designativo em inglês.

Podemos estruturar um registrador que permita o deslocamento para a esquerda. Se entrarmos com uma informação no registrador, teremos:

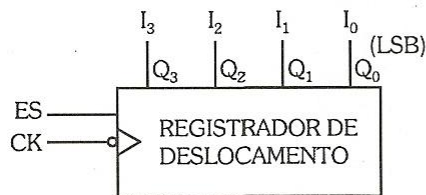



Figura 6.39

Se aplicarmos um deslocamento à esquerda, levando a saída Q_0 para 0, teremos a seguinte situação:

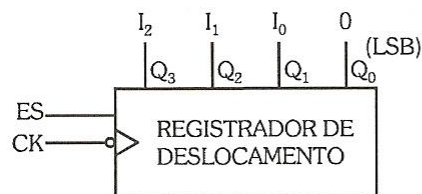



Figura 6.40

Podemos notar que essa operação significa multiplicarmos um número binário por 2. Para exemplificar, utilizaremos a informação:

$I = 0\ 0\ 0\ 1\ (1_{10})$

Registrador ⇨ $Q_3 = 0$, $Q_2 = 0$, $Q_1 = 0$ e $Q_0 = 1$

Se fizermos um deslocamento para a esquerda, teremos na saída, a seguinte situação:

$Q_3 = 0$, $Q_2 = 0$, $Q_1 = 1$ e $Q_0 = 0$

Notamos que a informação recolhida na saída, será:

$I = 0\ 0\ 1\ 0\ (2_{10})$

Podemos facilmente verificar que o número foi multiplicado por 2.

O deslocamento à esquerda é também conhecido como **Shift-Left**, termo designativo em inglês.

6.3.6 Exercícios Resolvidos

1 - A partir dos sinais aplicados às entradas, esboce as formas de onda das saídas para o Registrador de Deslocamento de 4 bits, visto na figura 6.41.

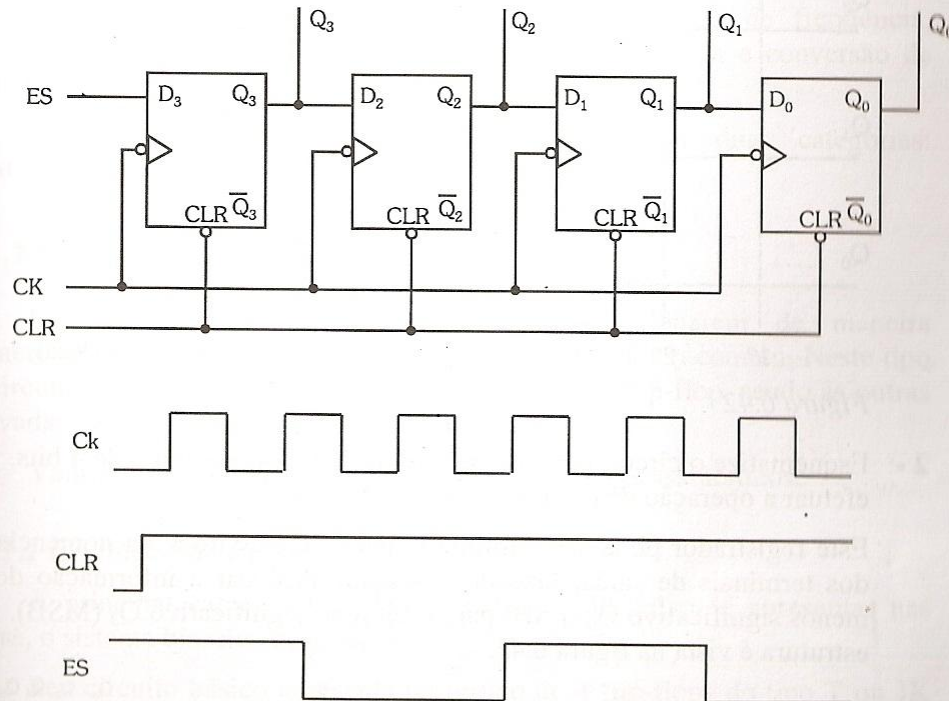



Figura 6.41

Para solucionar, vamos considerar cada descida de clock e verificar, a partir da entrada série (ES), o nível de saída registrado em cada bloco anterior. A figura 6.42 apresenta os sinais de saída resultantes deste processo, sendo o registrador inicialmente zerado pela forma de onda da entrada clear.

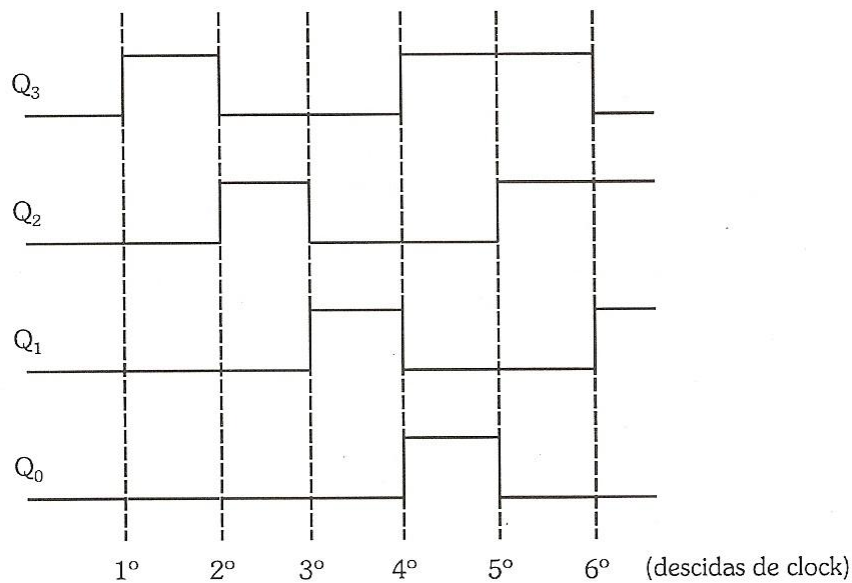



Figura 6.42

2 - Esquematize o circuito de um Registrador de Deslocamento de 4 bits, para efetuar a operação de deslocamento à esquerda.

Este registrador pode ser estruturado pela simples troca da nomenclatura dos terminais de saída, fazendo o sistema deslocar a informação do bit menos significativo Q_0 (LSB) para o bit mais significativo Q_3 (MSB). Esta estrutura é vista na figura 6.43.

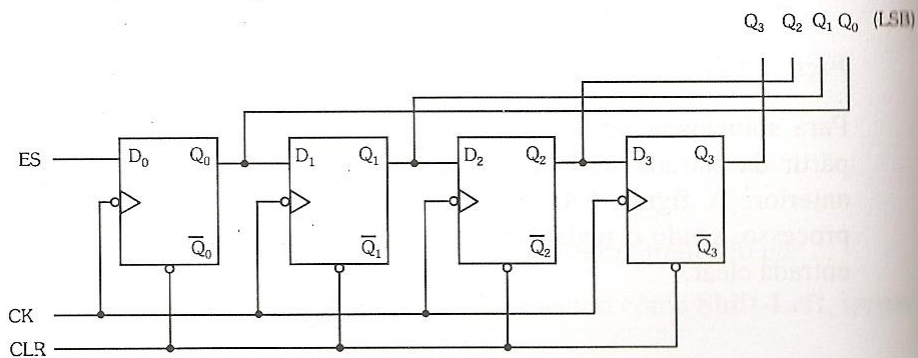



Figura 6.43

6.4 Contadores

Contadores são circuitos digitais que variam os seus estados, sob o comando de um clock, de acordo com uma seqüência predeterminada. São utilizados principalmente para contagens diversas, divisão de freqüência, medição de freqüência e tempo, geração de formas de onda e conversão de analógico para digital.

Basicamente, estes sistemas, são divididos em duas categorias: **Contadores Assíncronos e Síncronos.**

6.4.1 Contadores Assíncronos

São caracterizados por seus flip-flops funcionarem de maneira assíncrona (sem sincronismo), não tendo entradas clock em comum. Neste tipo de circuito, a entrada clock se faz apenas no primeiro flip-flop, sendo as outras derivadas das saídas dos blocos anteriores.

Vamos, a seguir, analisar os principais contadores assíncronos.

6.4.1.1 Contador de Pulsos

A principal característica de um contador de pulsos é apresentar nas saídas, o sistema binário em seqüência.

Seu circuito básico apresenta um grupo de 4 flip-flops do tipo T ou JK Mestre-Escravo, os quais possuem a entrada T ou, no caso, J e K iguais a 1, originando na saída $Qf = \overline{Q}a$, a cada descida de clock.

A entrada dos pulsos se faz através da entrada clock do 1º flip-flop, sendo as entradas clock dos flip-flops seguintes, conectadas às saídas Q dos respectivos antecessores conforme circuito visto na figura 6.44.

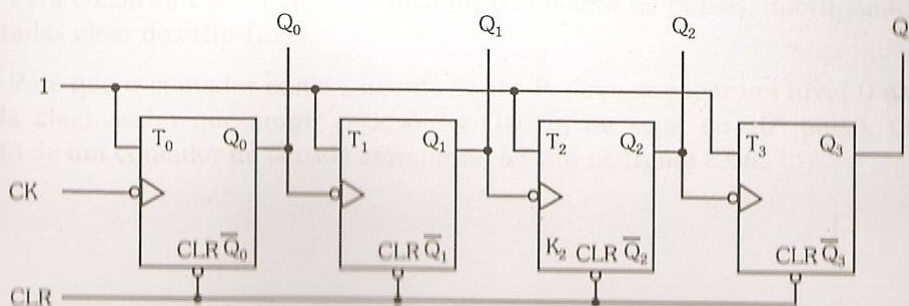



Figura 6.44

Vamos fazer, inicialmente, com que todos os flip-flops assumam saídas iguais a 0, através da aplicação de um nível 0 à entrada clear. A cada descida do pulso de clock, o 1º flip-flop irá mudar de estado, sendo esta troca aplicada à entrada do 2º flip-flop, fazendo com que este troque de estado a cada descida da saída Q_0, assim sucessivamente.

Vamos analisar este comportamento através da tabela 6.16.

Descidas de clock	Saídas Q_0	Q_1	Q_2	Q_3	
1ª	0	0	0	0	(Estado inicial, imposto por CLR = 0)
2ª	1	0	0	0	(Após a 1ª descida de clock: $Q_0=1$)
3ª	0	1	0	0	(Após a 2ª descida: $Q_0=0$ e $Q_1=1$, obtido pela descida de Q_0)
4ª	1	1	0	0	($Q_0=1$ e Q_1 permanece igual a1)
5ª	0	0	1	0	($Q_0=0 \Rightarrow Q_1=0 \Rightarrow Q_2=1$)
6ª	1	0	1	0	($Q_0=1$, Q_1 e Q_2 permanecem)
7ª	0	1	1	0	($Q_0=0 \Rightarrow Q_1=1$)
8ª	1	1	1	0	($Q_0=1$)
9ª	0	0	0	1	($Q_0=0 \Rightarrow Q_1=0 \Rightarrow Q_2=0 \Rightarrow Q_3=1$)
10ª	1	0	0	1	($Q_0=1$)
11ª	0	1	0	1	($Q_0=0 \Rightarrow Q_1=1$)
12ª	1	1	0	1	($Q_0=1$)
13ª	0	0	1	1	($Q_0=0 \Rightarrow Q_1=0 \Rightarrow Q_2=1$)
14ª	1	0	1	1	($Q_0=1$)
15ª	0	1	1	1	($Q_0=0 \Rightarrow Q_1=1$)
16ª	1	1	1	1	($Q_0=1$)
17ª	0	0	0	0	($Q_0=0 \Rightarrow Q_1=0 \Rightarrow Q_2=0 \Rightarrow Q_3=0$)

Tabela 6.16

Considerando Q_0 como bit menos significativo (LSB) e Q_3 como mais significativo (MSB), temos nas saídas o sistema binário em seqüência (0000 a 1111). Notamos ainda, que após a 16ª descida de clock, o contador irá reiniciar a contagem. A figura 6.45 apresenta toda a seqüência obtida graficamente, a partir da variação aplicada à entrada clock do sistema.

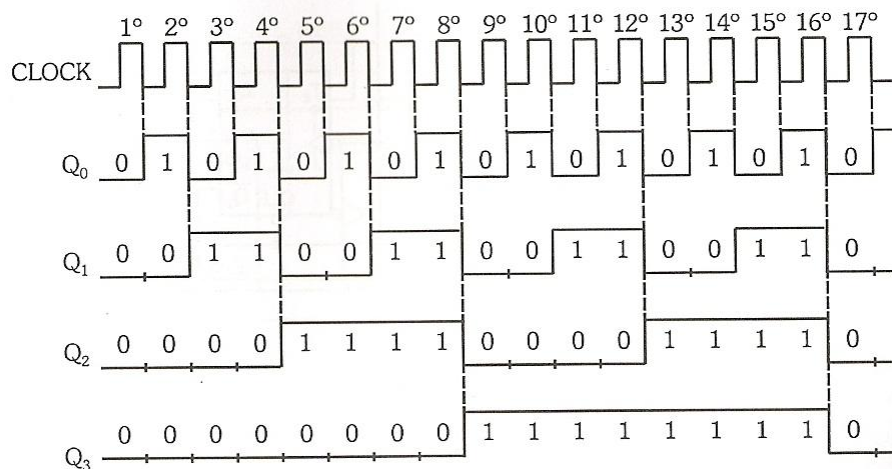



Figura 6.45

Analisando os gráficos, notamos que o período de Q_0 é o dobro do período do clock, logo, a freqüência de Q_0 será a metade da freqüência do clock, pois f = 1/T. Analisando a saída Q_1, veremos que seu período é o dobro de Q_0 e o quádruplo do clock, logo, sua freqüência será a metade de Q_0 e um quarto da freqüência do pulso de clock. Isto se estenderá sucessivamente aos demais flip-flops. Assim sendo, podemos notar que uma das aplicações dos contadores será a de dividir a freqüência de sinais (onda quadrada) aplicados à entrada clock. No caso deste contador, a divisão será por um número múltiplo de 2^N, onde N é o número de flip-flops utilizados.

6.4.1.2 Contador de Década

O contador de década é o circuito que efetua a contagem em números binários de 0 a 9_{10} (10 algarismos). Isso significa acompanhar a seqüência do código BCD 8421 de 0000 até 1001.

Para construir este circuito, utilizamos o contador de pulsos, interligando as entradas clear dos flip-flops.

Para que o contador conte somente de 0 a 9, deve-se jogar um nível 0 na entrada clear assim que surgir o caso 10 (1010), ou seja, no 10º pulso. O circuito de um contador de década assíncrono é visto na figura 6.46.

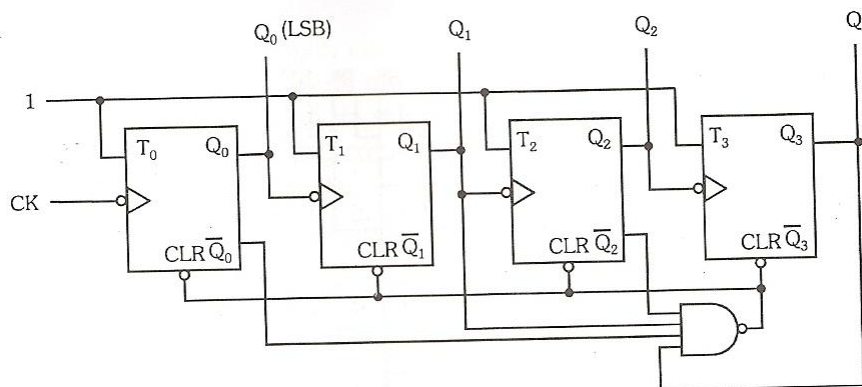



Figura 6.46

Temos, neste caso, a seguinte tabela da verdade:

Descidas de clock	Q_3	Q_2	Q_1	Q_0	CLR
1ª	0	0	0	0	1
2ª	0	0	0	1	1
3ª	0	0	1	0	1
4ª	0	0	1	1	1
5ª	0	1	0	0	1
6ª	0	1	0	1	1
7ª	0	1	1	0	1
8ª	0	1	1	1	1
9ª	1	0	0	0	1
10ª	1	0	0	1	1
	1	0	1	0	0

Tabela 6.17

Após a 10ª descida de clock, o contador tende a assumir o estado $Q_0 = 0$, $Q_1 = 1$, $Q_2 = 0$, $Q_3 = 1$ (1010_2), porém, neste instante, a entrada clear vai para 0, zerando o contador, ou seja, fazendo com que assuma o estado 0 (0000), reiniciando a contagem.

Uma outra forma de obter o mesmo clear ou reset no caso 1010, utilizando uma porta NE com menos entradas, consiste em ligarmos apenas Q_3 e Q_1 nesta, pois só serão iguais a 1 simultaneamente neste caso, zerando as saídas do mesmo jeito. A figura 6.47 ilustra este procedimento.

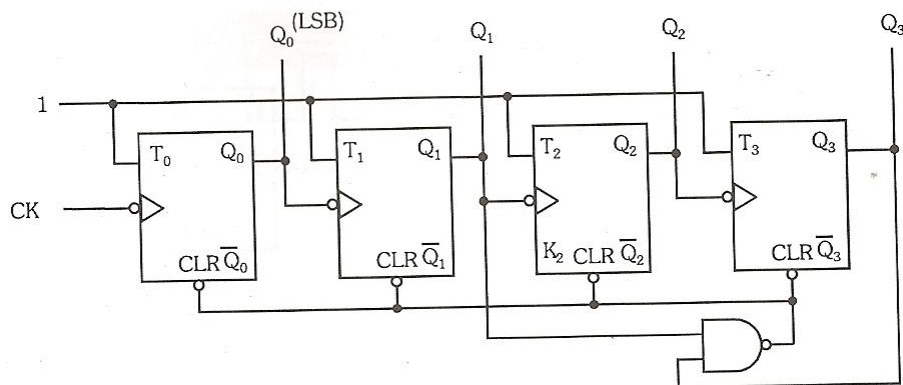



Figura 6.47

Este contador poderá ser utilizado como divisor de freqüência por 10 para uma onda quadrada aplicada à entrada clock, pois possui 10 estados de saída.

6.4.1.3 Contador Seqüencial de 0 a n

Vimos no item anterior, um contador que faz a contagem de 0 até 9_{10}. Utilizando o mesmo processo, podemos fazer um contador contar de 0 até um número n qualquer. Para isso, basta apenas verificarmos quais as saídas do contador para o caso seguinte a n, colocarmos estas saídas numa porta NE e à saída desta ligarmos as entradas clear dos flip-flops.

Para exemplificar, vamos elaborar o circuito de um contador de 0 a 5_{10}. Nesse caso, desejamos que o contador recomece a contagem após o estado 5, ou seja, passe para 0 todos os flip-flops.

Neste caso, o estado seguinte a n será o 6, ocasionando nas saídas: $Q_2 = 1$, $Q_1 = 1$ e $Q_0 = 0$ (110). Quando ocorrer então, deverá haver um 0 nas entradas clear interligadas, levando o contador a 0. Devemos, para tanto, ter na entrada da porta NE, a ligação de Q_2 e Q_1, pois na seqüência da contagem, estas irão assumir níveis 1 simultaneamente apenas no caso 110. A figura 6.48 mostra o circuito assim configurado.

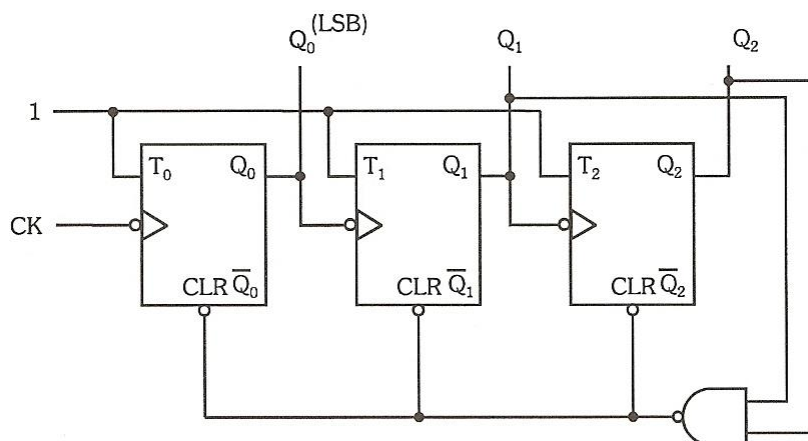



Figura 6.48

No circuito, utilizamos somente 3 flip-flops, pois são suficientes para contarmos de 0 a 7_{10} ($2^3 = 8$).

6.4.1.4 Contadores Assíncronos Decrescentes

Como vimos no item 6.4, os contadores se dividem em síncronos e assíncronos. Esta classificação é feita de acordo com a operação do clock do sistema.

Os contadores podem também ser classificados pelo tipo de contagem que executam, ou seja, se executam contagem crescente ou decrescente. A estes contadores damos os nomes de **contadores crescentes** e **contadores decrescentes** respectivamente.

Os contadores vistos até aqui são contadores crescentes, pois contam os números progressivamente de 0 a n.

Vamos estudar, agora, os contadores que efetuam a contagem decrescente. Esta, é vista na tabela 6.18.

Decimal	Binário			
15	1	1	1	1
14	1	1	1	0
13	1	1	0	1
12	1	1	0	0

Decimal	Binário			
11	1	0	1	1
10	1	0	1	0
9	1	0	0	1
8	1	0	0	0
7	0	1	1	1
6	0	1	1	0
5	0	1	0	1
4	0	1	0	0
3	0	0	1	1
2	0	0	1	0
1	0	0	0	1
0	0	0	0	0

Tabela 6.18

O circuito que efetua a contagem decrescente é o mesmo circuito que efetua a contagem crescente, com a única diferença de extrairmos as saídas dos terminais $\overline{Q}_0$, $\overline{Q}_1$, $\overline{Q}_2$ e $\overline{Q}_3$, sendo o terminal $\overline{Q}_0$, o bit menos significativo. Podemos notar, pela tabela da verdade, que a contagem decrescente nada mais é que o complemento da contagem crescente. Assim sendo, o circuito é visto na figura 6.49.

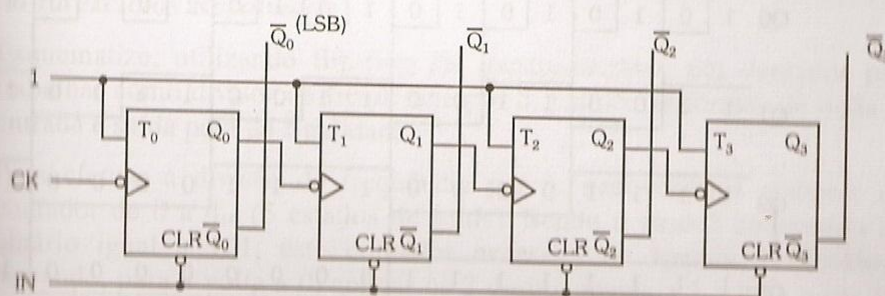



Figura 6.49

O estado inicial (1111) pode ser obtido pela aplicação de nível 0 na entrada IN, que irá zerar todos os flip-flops nas saídas Q, porém irá impor níveis 1 nas saídas $\overline{Q}$.

Um outro modo de montar um contador decrescente é injetando nas entradas clock dos flip-flops, as saídas complementares como é mostrado na figura 6.50.

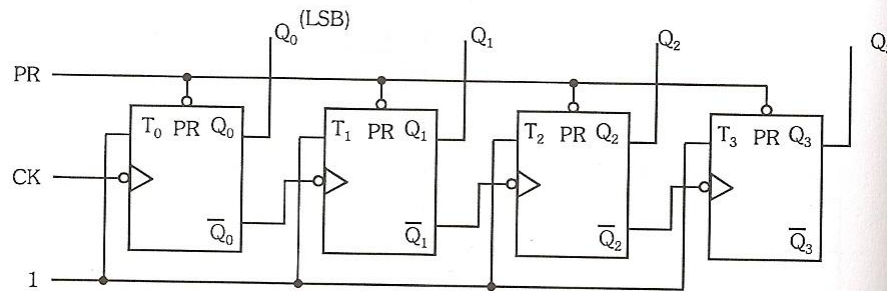



Figura 6.50

Neste circuito, os clocks dos flip-flops são, respectivamente, $\overline{Q}_0$, $\overline{Q}_1$ e $\overline{Q}_2$, logo Q_1, Q_2 e Q_3 irão trocar de estado nas subidas de Q_0, Q_1 e Q_2, respectivamente (descidas de $\overline{Q}_0$, $\overline{Q}_1$ e $\overline{Q}_2$), originando a contagem decrescente. O estado inicial pode ser obtido pela passagem da entrada PR para 0, estabelecendo nível 1 à saída de todos os flip-flops. A figura 6.51 mostra todas as formas de onda do sistema, desde a aplicação de uma onda quadrada à entrada clock.

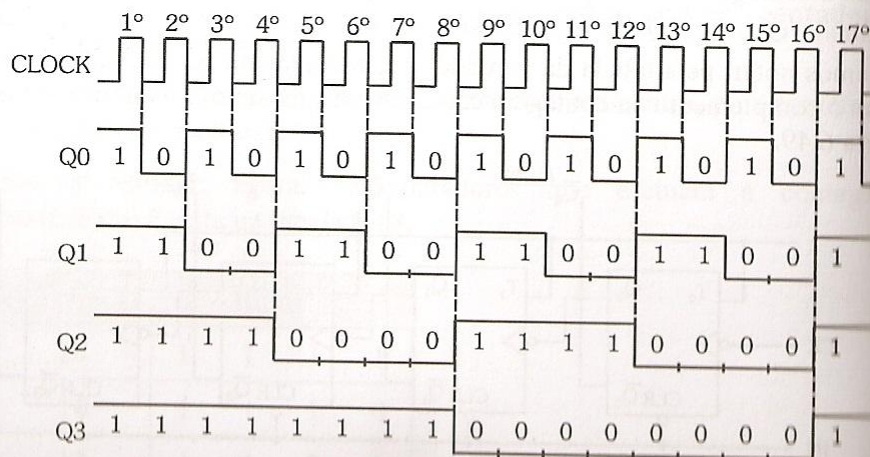



Figura 6.51

6.4.1.5 *Contador Assíncrono Crescente/Decrescente*

Podemos construir um contador que execute a contagem crescente ou decrescente. Para isso, utilizamos uma variável de controle que quando assume 1, faz o circuito executar contagem crescente e quando assume 0, faz a contagem decrescente. Este circuito é mostrado na figura 6.52.

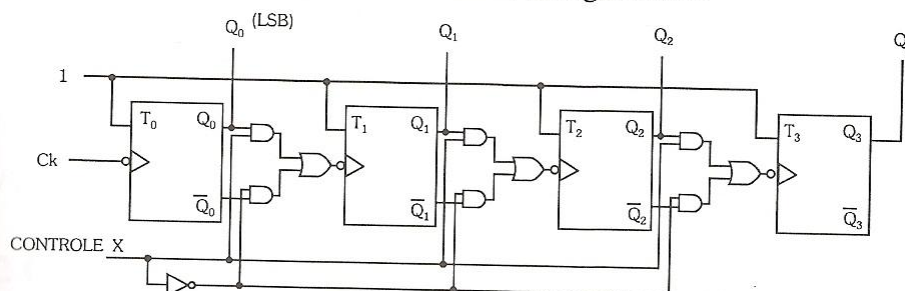



Figura 6.52

Notamos que, no circuito, quando o controle X estiver em 1, as saídas $\overline{Q}_0$, $\overline{Q}_1$ e $\overline{Q}_2$ estarão bloqueadas, fazendo com que entrem as saídas Q_0, Q_1 e Q_2, nas entradas clock dos flip-flops respectivamente. Isto fará com que o contador conte crescentemente.

Quando o controle X estiver em 0, a situação inverter-se-á e, por conseguinte, o contador contará decrescentemente.

Notamos, ainda, que Q_0 será a saída do bit menos significativo (LSB).

O contador crescente/decrescente é também denominado **Up/Down Counter**, que é o termo designativo em inglês.

6.4.1.6 *Exercícios Resolvidos*

1 - Esquematize, utilizando flip-flop JK mestre-escravo, um contador para trabalhar como divisor de freqüência por 5. Esboce as formas de onda da entrada e saída para tal finalidade.

Para efetuar a divisão de freqüência por 5, necessitamos elaborar um contador de 0 a 4_{10} (5 estados de saída). Sendo o caso 5 convertido em binário igual a 101, este contador necessita de apenas 3 Flip-flops, bastando, para a volta a 0, aplicar os 2 dígitos iguais a 1 a uma porta NE ligada às entradas clear, pois, estes valerão 1 simultaneamente pela primeira vez na seqüência de 3 bits. O contador assim disposto, é visto na figura 6.53.

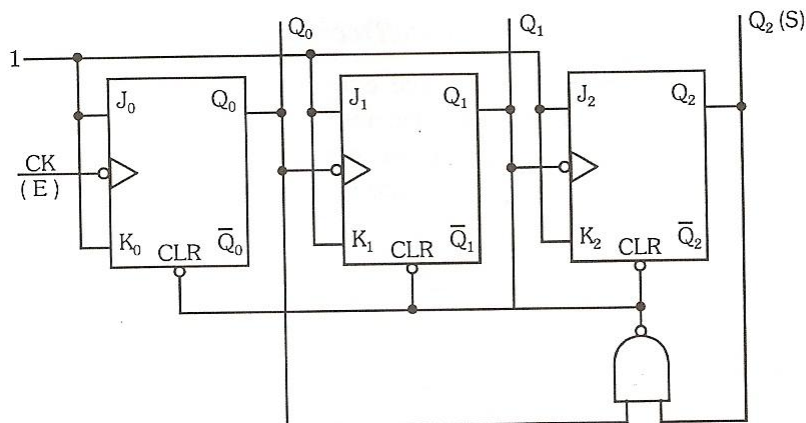



Figura 6.53

A forma de onda de entrada (E) é aplicada à entrada clock do primeiro flip-flop, sendo obtida através do terminal Q_2 do último flip-flop. Estas formas de onda são vistas na figura 6.54.

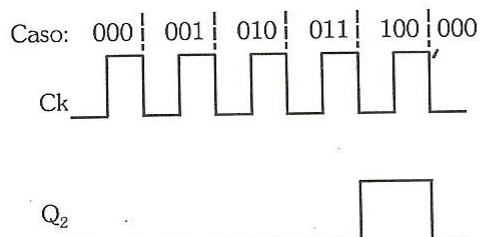



Figura 6.54

Notamos pela figura, que embora a forma de onda do Q_2 não seja simétrica, o seu período será 5 vezes maior em relação à aplicada em clock.

2 - Elabore um contador decrescente de 7_{10} a 0. O circuito deve possuir um terminal que, quando aterrado, estabelece o caso inicial.

Este circuito pode ser obtido por 3 flip-flops (111 a 000), sendo as entradas clock do 2º e 3º blocos acionadas pelas saídas $\overline{Q}$ das anteriores. Para estabelecer o caso inicial, basta interligarmos as entradas preset dos flip-flops. O circuito, configurado com flip-flops JK mestre-escravo é visto na figura 6.55.

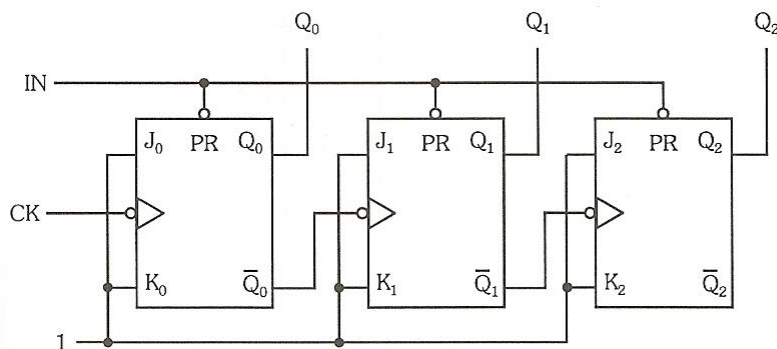



Figura 6.55

6.4.2 Contadores Síncronos

Estes contadores possuem entradas clock curto-circuitadas, ou seja, o clock entra em todos os flip-flops simultaneamente, fazendo todos atuarem de forma sincronizada.

Para que haja mudanças de estado, devemos então estudar o comportamento das entradas J e K dos vários flip-flops, para que tenhamos nas saídas, as seqüências desejadas.

Para estudarmos os contadores síncronos devemos sempre escrever a tabela da verdade, estudando quais devem ser as entradas J e K dos vários flip-flops, para que estes assumam o estado seguinte. Para isso, vamos utilizar a tabela da verdade do flip-flop JK:

J	K	Qf	
0	0	Qa	(mantém o estado)
0	1	0	(fixa 0)
1	0	1	(fixa 1)
1	1	$\overline{Qa}$	(inverte o estado)

Tabela 6.19

A partir desta tabela, construímos outra, relacionando os estados de saída e as entradas J e K:

	Qa	Qf	J	K
1)	0	0	0	X
2)	0	1	1	X
3)	1	0	X	1
4)	1	1	X	0

Tabela 6.20

Vamos, a seguir, analisar cada caso:

1) Se o flip-flop estiver em 0 (Qa = 0) e quisermos que o estado a ser assumido seja 0 (Qf = 0), podemos tanto manter o estado do flip-flop (J = 0, K = 0 ⇨ Qf = Qa), como fixar 0 (J = 0, K = 1 ⇨ Qf = 0), logo, se J = 0 e K = X, teremos a passagem de Qa = 0 para Qf = 0.

2) Se o flip-flop estiver em 0 (Qa = 0) e quisermos que o estado a ser assumido seja 1 (Qf = 1), podemos tanto inverter o estado (J = 1, K = 1 ⇨ Qf = $\overline{Qa}$), como fixarmos 1 (J = 1, K = 0 ⇨ Qf = 1), logo, se J = 1 e K = X, teremos a passagem de Qa = 0 para Qf = 1.

3) Quando o flip-flop estiver em 1 (Qa = 1) e quisermos que ele vá para 0 (Qf = 0), podemos inverter o estado (J = 1, K = 1 ⇨ Qf = $\overline{Qa}$) ou fixar 0 (J = 0, K = 1 ⇨ Qf = 0), logo, se J = X e K = 1, teremos a passagem de Qa = 1 para Qf = 0.

4) Quando o flip-flop estiver em 1 (Qa = 1) e quisermos que ele permaneça em 1 (Qf = 1), podemos manter o estado (J = 0, K = 0 ⇨ Qf = Qa) ou fixarmos 1 (J = 1, K = 0 ⇨ Qf = 1), logo, se J = X e K = 0, teremos a passagem de Qa = 1 para Qf = 1.

De posse dos resultados das entradas J e K dos flip-flops para a seqüência desejada, obtidos da tabela, efetuamos as simplificações e montamos um circuito combinacional que em função das saídas dos flip-flops irá atuar nestas entradas para processar as mudanças de estado.

Genericamente, um contador síncrono possui o esquema visto na figura 6.56.

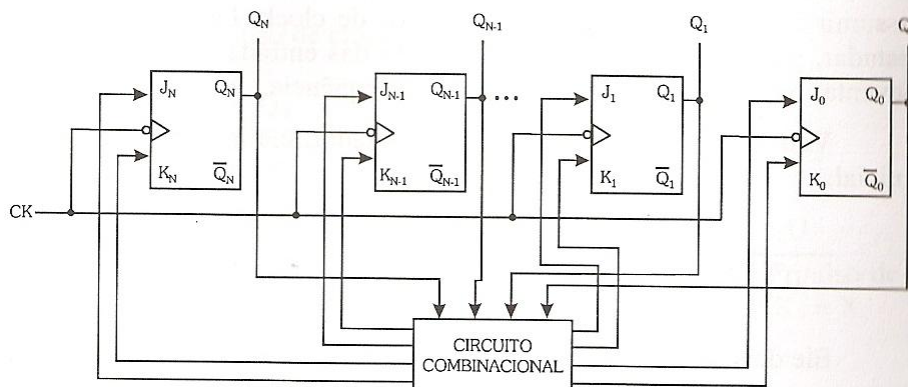



Figura 6.56

Nos próximos itens, vamos, a título de exemplo, efetuar projetos de contadores síncronos para gerar seqüências conhecidas ou de grande aplicabilidade na prática.

6.4.2.1 Contador Síncrono Gerador de Código Binário de 4 Bits

Para gerarmos esse código, necessitamos de 4 flip-flops JK mestre-escravo, ou seja, um flip-flop para cada bit do código. A tabela 6.21 apresenta a seqüência proposta.

ck	Q_3	Q_2	Q_1	Q_0
1ª	0	0	0	0
2ª	0	0	0	1
3ª	0	0	1	0
4ª	0	0	1	1
5ª	0	1	0	0
6ª	0	1	0	1
7ª	0	1	1	0
8ª	0	1	1	1
9ª	1	0	0	0
10ª	1	0	0	1
11ª	1	0	1	0
12ª	1	0	1	1
13ª	1	1	0	0
14ª	1	1	0	1
15ª	1	1	1	0
16ª	1	1	1	1

Tabela 6.21

Esta tabela apresenta a seqüência que as saídas dos flip-flops devem assumir em função da presença de pulsos de clock. Para o projeto, devemos estudar, para cada caso, o comportamento das entradas J e K dos flip-flops e levantar o circuito necessário para gerar a seqüência.

Vamos supor que ao ligarmos o contador, ele assuma o seguinte estado inicial:

Q_3	Q_2	Q_1	Q_0
0	0	0	0

Ele deverá, após o 1º pulso de clock, passar para o estado seguinte:

Q_3	Q_2	Q_1	Q_0
0	0	0	1

Sob a presença do 1º pulso de clock, temos:

⇨ Q_3 : estava em 0, deve passar para 0, logo, antes do 1º pulso de clock, devemos ter as seguintes entradas neste flip-flop: $J_3 = 0$ e $K_3 = X$ ($J = 0$ e $K = X$ ⇨ $Qa = 0$ passa para $Qf = 0$).

⇨ Q_2 : caso análogo a Q_3 , logo $J_2 = 0$ e $K_2 = X$.

⇨ Q_1: idem, logo, $J_1 = 0$ e $K_1 = X$.

⇨ Q_0: estava em 0, após o 1º pulso de clock deve mudar para 1, logo antes do 1º pulso de clock, devemos ter as seguintes entradas neste flip-flop: $J_0 = 1$ e $K_0 = X$ ($J = 1$ e $K = X$ ⇨ $Qa = 0$ passa para $Qf = 1$).

Podemos, a partir da análise, escrever a primeira linha da tabela da verdade.

Descida do pulso de clock	Q_3 Q_2 Q_1 Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
1ª	0 0 0 0	0	X	0	X	0	X	1	X
	0 0 0 1								

O contador está, agora, no estado:

Q_3	Q_2	Q_1	Q_0
0	0	0	1

E deve, após o 2º pulso de clock, passar para:

Q_3	Q_2	Q_1	Q_0
0	0	1	0

Vamos, então, analisar as entradas J e K para este caso:

- ⇨ Q_3: estava em 0 e deve permanecer em 0, logo, antes do 2º pulso de clock, devemos ter a seguinte situação de entrada: $J_3 = 0$ e $K_3 = X$
- ⇨ Q_2 : possui caso análogo a Q_3 , logo $J_2 = 0$ e $K_2 = X$.
- ⇨ Q_1: estava em 0 e deve passar para 1, logo, antes do 2º pulso de clock, devemos ter a seguinte situação de entrada no flip-flop: $J_1 = 1$ e $K_1 = X$.
- ⇨ Q_0 : estava em 1 e deve passar para 0, logo antes do 2º pulso de clock, devemos ter a seguinte situação de entrada: $J_0 = X$ e $K_0 = 1$.

Podemos, a partir disso, escrever a segunda linha da tabela da verdade:

Descida do pulso de clock	Q_3 Q_2 Q_1 Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
1ª	0 0 0 0	0	X	0	X	0	X	1	X
2ª	0 0 0 1	0	X	0	X	1	X	X	1
	0 0 1 0								

Para fixarmos melhor o procedimento, vamos analisar mais uma mudança do contador, ou seja, após a descida do 3º pulso de clock, a passagem do estado 2 para o estado 3:

	Q_3	Q_2	Q_1	Q_0
estado 2 ⟶	0	0	1	0
estado 3 ⟶	0	0	1	1

- ⇨ Q_3 : vai de 0 para 0 ⇨ $J_3 = 0$ e $K_3 = X$
- ⇨ Q_2 : vai de 0 para 0 ⇨ $J_2 = 0$ e $K_2 = X$
- ⇨ Q_1: vai de 1 para 1 ⇨ $J_1 = X$ e $K_1 = 0$
- ⇨ Q_0: vai de 0 para 1 ⇨ $J_0 = 1$ e $K_0 = X$

A tabela da verdade, até a terceira linha, será:

Descidas do pulso de clock	Q_3 Q_2 Q_1 Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
1ª	0 0 0 0	0	X	0	X	0	X	1	X
2ª	0 0 0 1	0	X	0	X	1	X	X	1
3ª	0 0 1 0	0	X	0	X	X	0	1	X
	0 0 1 1								

Utilizando o mesmo procedimento para os outros casos, obtemos a tabela completa:

Descidas do pulso de clock	Q_3 Q_2 Q_1 Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
1ª	0 0 0 0	0	X	0	X	0	X	1	X
2ª	0 0 0 1	0	X	0	X	1	X	X	1
3ª	0 0 1 0	0	X	0	X	X	0	1	X
4ª	0 0 1 1	0	X	1	X	X	1	X	1
5ª	0 1 0 0	0	X	X	0	0	X	1	X
6ª	0 1 0 1	0	X	X	0	1	X	X	1
7ª	0 1 1 0	0	X	X	0	X	0	1	X
8ª	0 1 1 1	1	X	X	1	X	1	X	1
9ª	1 0 0 0	X	0	0	X	0	X	1	X
10ª	1 0 0 1	X	0	0	X	1	X	X	1
11ª	1 0 1 0	X	0	0	X	X	0	1	X
12ª	1 0 1 1	X	0	1	X	X	1	X	1
13ª	1 1 0 0	X	0	X	0	0	X	1	X
14ª	1 1 0 1	X	0	X	0	1	X	X	1
15ª	1 1 1 0	X	0	X	0	X	0	1	X
16ª	1 1 1 1	X	1	X	1	X	1	X	1

Tabela 6.22

Notamos que, no projeto, o estado 0 foi considerado após o estado 15, pois ao final, o contador deve reiniciar a contagem.

Para obter as expressões de J_3, K_3, J_2, K_2, J_1, K_1, J_0 e K_0, simplificadas, vamos utilizar diagramas de Veitch-Karnaugh:

J_3:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	0	0	0
$\overline{Q}_3 Q_2$	0	0	1	0
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	X	X	X	X

(a)

$J_3 = Q_2\, Q_1\, Q_0$

K_3:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	X	X	X
$\overline{Q}_3 Q_2$	X	X	X	X
$Q_3 Q_2$	0	0	1	0
$Q_3\overline{Q}_2$	0	0	0	0

(b)

$K_3 = Q_2\, Q_1\, Q_0$

$\therefore\ J_3 = K_3 = Q_2\, Q_1\, Q_0$

J_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	0	1	0
$\overline{Q}_3 Q_2$	X	X	X	X
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	0	0	1	0

(c)

$J_2 = Q_1\, Q_0$

K_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	X	X	X
$\overline{Q}_3 Q_2$	0	0	1	0
$Q_3 Q_2$	0	0	1	0
$Q_3\overline{Q}_2$	X	X	X	X

(d)

$K_2 = Q_1\, Q_0$

$\therefore\ J_2 = K_2 = Q_1\, Q_0$

Figura 6.57 (parte)

J_1:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1Q_0$	Q_1Q_0	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	1	X	X
$\overline{Q}_3Q_2$	0	1	X	X
Q_3Q_2	0	1	X	X
$Q_3\overline{Q}_2$	0	1	X	X

(e)

$J_1 = Q_0$

K_1:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1Q_0$	Q_1Q_0	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	X	1	0
$\overline{Q}_3Q_2$	X	X	1	0
Q_3Q_2	X	X	1	0
$Q_3\overline{Q}_2$	X	X	1	0

(f)

$K_1 = Q_0$

$\therefore J_1 = K_1 = Q_0$

J_0:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1Q_0$	Q_1Q_0	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	1	X	X	1
$\overline{Q}_3Q_2$	1	X	X	1
Q_3Q_2	1	X	X	1
$Q_3\overline{Q}_2$	1	X	X	1

(g)

$J_0 = 1$

K_0:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1Q_0$	Q_1Q_0	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	1	1	X
$\overline{Q}_3Q_2$	X	1	1	X
Q_3Q_2	X	1	1	X
$Q_3\overline{Q}_2$	X	1	1	X

(h)

$K_0 = 1$

$\therefore J_0 = K_0 = 1$

Figura 6.57

O circuito completo deste contador é visto na figura 6.58.

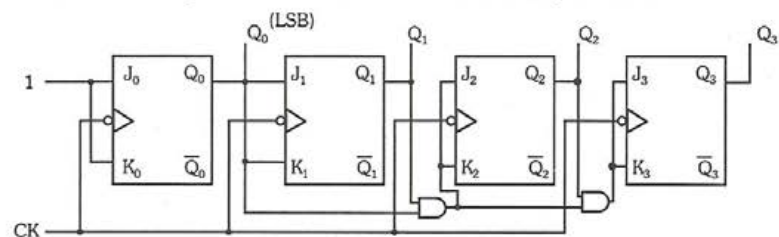



Figura 6.58

As entradas clear e preset dos flip-flops poderiam, da mesma forma que nos contadores assíncronos, ser utilizadas para estabelecer o caso inicial, zerar o contador, ou ainda, fixar qualquer caso no decorrer da contagem.

6.4.2.2 Contador de Década

Vamos construir um contador de década síncrono. Para isso, utilizaremos o mesmo processo já visto.

Primeiramente, vamos verificar o comportamento das entradas J e K:

Q_3 Q_2 Q_1 Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0 0 0 0	0	X	0	X	0	X	1	X
0 0 0 1	0	X	0	X	1	X	X	1
0 0 1 0	0	X	0	X	X	0	1	X
0 0 1 1	0	X	1	X	X	1	X	1
0 1 0 0	0	X	X	0	0	X	1	X
0 1 0 1	0	X	X	0	1	X	X	1
0 1 1 0	0	X	X	0	X	0	1	X
0 1 1 1	1	X	X	1	X	1	X	1
1 0 0 0	X	0	0	X	0	X	1	X
1 0 0 1	X	1	0	X	0	X	X	1

Tabela 6.23

Supondo conseguir o estado inicial através das entradas clear, vamos considerar os estados não pertencentes à seqüência como irrelevantes.

Vamos, agora, transpor os casos para os diagramas e simplificar:

$J_3 = Q_2 Q_1 Q_0$

$K_3 = Q_0$

J_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	0	1	0
$\overline{Q}_3 Q_2$	X	X	X	X
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	0	0	X	X

(c)

$J_2 = Q_1\,Q_0$

K_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	X	X	X
$\overline{Q}_3 Q_2$	0	0	1	0
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	X	X	X	X

(d)

$K_2 = Q_1\,Q_0$

J_1:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	1	X	X
$\overline{Q}_3 Q_2$	0	1	X	X
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	0	0	X	X

(e)

$J_1 = Q_0\,\overline{Q}_3$

K_1:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	X	1	0
$\overline{Q}_3 Q_2$	X	X	1	0
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	X	X	X	X

(f)

$K_1 = Q_0$

J_0:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	1	X	X	1
$\overline{Q}_3 Q_2$	1	X	X	1
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	1	X	X	X

(g)

$J_0 = 1$

K_0:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	1	1	X
$\overline{Q}_3 Q_2$	X	1	1	X
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	X	1	X	X

(h)

$K_0 = 1$

Figura 6.59

Vamos, mediante as expressões obtidas, esquematizar o circuito do contador de década síncrono:

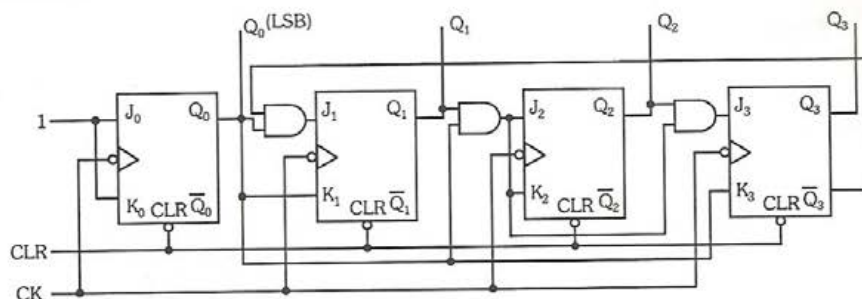



Figura 6.60

6.4.2.3 Contador Gerador de uma Seqüência Qualquer

Podemos construir um contador que gere uma seqüência qualquer. Para isso, basta estabelecermos a seqüência e seguirmos o método já conhecido, ou seja, o da determinação das entradas J e K. Os estados que não fizerem parte da seqüência deverão ser considerados como condições irrelevantes, ou ser encadeados objetivando atingir o estado inicial.

Para exemplificarmos, vamos construir um contador que gere a seguinte seqüência: 0 ⇨ 1 ⇨ 2 ⇨ 3 ⇨ 10 ⇨ 13 ⇨ 0.

O **loop** que o contador deve efetuar para acompanhar a seqüência é visto no **diagrama de estados** visto na figura 6.61.

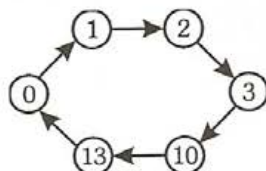



Figura 6.61

Notamos que os estados que não pertencem à seqüência são: 4, 5, 6, 7, 8, 9, 11, 12, 14 e 15. Vamos fazer, então, com que o contador, estando no estado 4, após o pulso de clock, vá para o estado 5, deste para o 6 e assim sucessivamente, até que do estado 15 vá para 0 que inicia a seqüência. Esquematicamente, temos:

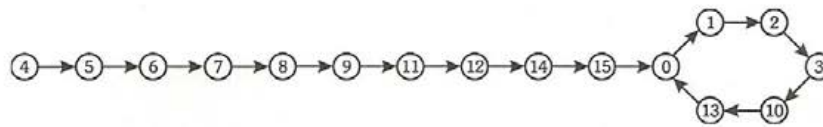



Figura 6.62

Notamos que este contador, na pior das hipóteses (no estado 4), irá entrar no loop da seqüência após o 10º pulso de clock.

Vamos, agora, montar a tabela da verdade:

Estados	Q_3	Q_2	Q_1	Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
(0)	0	0	0	0	0	X	0	X	0	X	1	X
(1)	0	0	0	1	0	X	0	X	1	X	X	1
(2)	0	0	1	0	0	X	0	X	X	0	1	X
(3)	0	0	1	1	1	X	0	X	X	0	X	1
(10)	1	0	1	0	o estado 3 antecede o 10							
(4)	0	1	0	0	0	X	X	0	0	X	1	X
(5)	0	1	0	1	0	X	X	0	1	X	X	1
(6)	0	1	1	0	0	X	X	0	X	0	1	X
(7)	0	1	1	1	1	X	X	1	X	1	X	1
(8)	1	0	0	0	X	0	0	X	0	X	1	X
(9)	1	0	0	1	X	0	0	X	1	X	X	0
(11)	1	0	1	1	o estado 9 antecede o 11							
(10)	1	0	1	0	X	0	1	X	X	1	1	X
(13)	1	1	0	1	o estado 10 antecede o 13							
(11)	1	0	1	1	X	0	1	X	X	1	X	1
(12)	1	1	0	0	X	0	X	0	1	X	0	X
(14)	1	1	1	0	o estado 12 antecede o 14							
(13)	1	1	0	1	X	1	X	1	0	X	X	1
(0)	0	0	0	0	o estado 13 antecede o 0							
(14)	1	1	1	0	X	0	X	0	X	0	1	X
(15)	1	1	1	1	X	1	X	1	X	1	X	1
(0)	0	0	0	0	o estado 15 antecede o 0							

Tabela 6.24

Vamos, agora, mediante a utilização dos diagramas, obter as expressões simplificadas das entradas J e K dos flip-flops:

J_3:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	0	1	0
$\overline{Q}_3 Q_2$	0	0	1	0
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	X	X	X	X

(a)

$J_3 = Q_1 Q_0$

K_3:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	X	X	X	X
$\overline{Q}_3 Q_2$	X	X	X	X
$Q_3 Q_2$	0	1	1	0
$Q_3\overline{Q}_2$	0	0	0	0

(b)

$K_3 = Q_2 Q_0$

J_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3\overline{Q}_2$	0	0	0	0
$\overline{Q}_3 Q_2$	X	X	X	X
$Q_3 Q_2$	X	X	X	X
$Q_3\overline{Q}_2$	0	0	1	1

(c)

$J_2 = Q_3 Q_1$

K_2:

(d)

$K_2 = Q_3 Q_0 + Q_1 Q_0$

Figura 6.63 (parte)

J_1:

(e)

$$J_1 = Q_0\,\overline{Q}_3 + Q_0\,\overline{Q}_2 + Q_3\,Q_2\,\overline{Q}_0$$

$$J_1 = Q_0\,(\overline{Q}_3 + \overline{Q}_2) + \overline{Q}_0\,(Q_3\,Q_2)$$

$$J_1 = Q_0\,(\overline{Q_3 Q_2}\,) + \overline{Q}_0\,(Q_3\,Q_2)$$

$$J_1 = Q_0 \oplus (Q_3\,Q_2)$$

(f)

$$K_1 = Q_2\,Q_0 + Q_3\,\overline{Q}_2$$

J_0:

(g)

$$J_0 = \overline{Q}_3 + \overline{Q}_2 + Q_1$$

K_0:

(h)

$$K_0 = \overline{Q}_3 + Q_2 + Q_1$$

Figura 6.63

O circuito obtido das expressões é visto na figura 6.64.

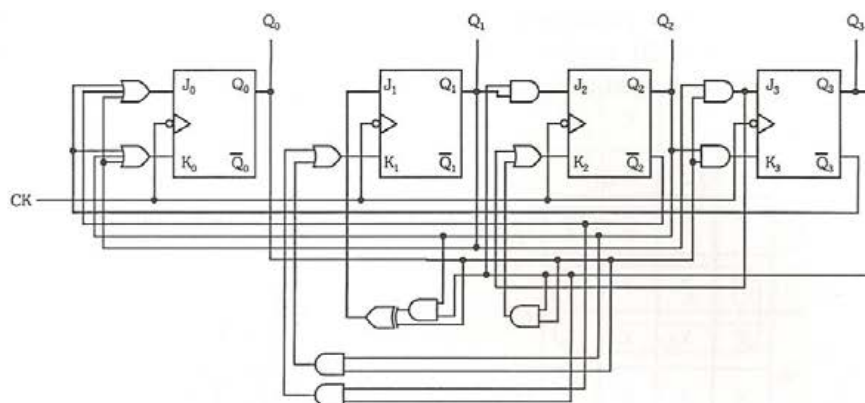



Figura 6.64

6.4.2.4 Contador de Anel

Este contador, também conhecido em inglês como **Ring Counter**, irá gerar a seqüência da tabela 6.25.

Q_3	Q_2	Q_1	Q_0
0	0	0	1
0	0	1	0
0	1	0	0
1	0	0	0

Tabela 6.25

Vamos levantar, de modo análogo aos anteriores, o comportamento das entradas J e K, perante a seqüência apresentada. Para isso, montamos a tabela da verdade:

Q_3 Q_2 Q_1 Q_0	J_3	K_3	J_2	K_2	J_1	K_1	J_0	K_0
0 0 0 1	0	X	0	X	1	X	X	1
0 0 1 0	0	X	1	X	X	1	0	X
0 1 0 0	1	X	X	1	0	X	0	X
1 0 0 0	X	1	0	X	0	X	1	X

Tabela 6.26

Se obtivermos o estado inicial através das entradas preset e clear, faremos o contador permanecer sempre no loop da seqüência, logo, os outros estados tornar-se-ão irrelevantes. Podemos, então, transcrever a tabela da verdade para os diagramas:

J_3:

		$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3$	$\overline{Q}_2$	X	0	X	0
$\overline{Q}_3$	Q_2	1	X	X	X
Q_3	Q_2	X	X	X	X
Q_3	$\overline{Q}_2$	X	X	X	X

(a)

$J_3 = Q_2$

K_3:

		$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3$	$\overline{Q}_2$	X	X	X	X
$\overline{Q}_3$	Q_2	X	X	X	X
Q_3	Q_2	X	X	X	X
Q_3	$\overline{Q}_2$	1	X	X	X

(b)

$K_3 = \overline{Q}_2$ *

* Embora pudéssemos ligar a entrada K_3 em 1 (agrupamento máximo), podemos também ligá-la à saída $\overline{Q}_2$ (agrupamento da oitava $\overline{Q}_2$).

J_2:

(c)

$J_2 = Q_1$

K_2:

		$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{Q}_3$	$\overline{Q}_2$	X	X	X	X
$\overline{Q}_3$	Q_2	1	X	X	X
Q_3	Q_2	X	X	X	X
Q_3	$\overline{Q}_2$	X	X	X	X

(d)

$K_2 = \overline{Q}_1$ *

Figura 6.65 (Parte)

J_1:

(e)

$J_1 = Q_0$

K_1:

(f)

$K_1 = \overline{Q}_0$ *

J_0:

(g)

$J_0 = Q_3$

K_0:

(h)

$K_0 = \overline{Q}_3$ *

* K_2, K_1 e K_0, análogos a K_3.

Figura 6.65

Após obter as expressões, vamos esquematizar o circuito do contador em anel:

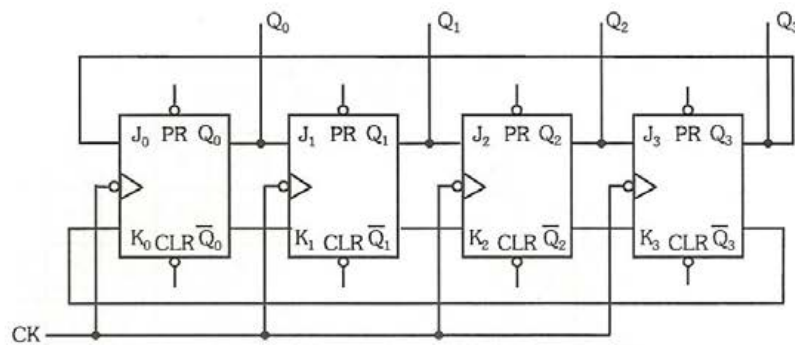



Figura 6.66

6.4.2.5 Contador Johnson

O circuito do contador Johnson é visto na figura 6.67.

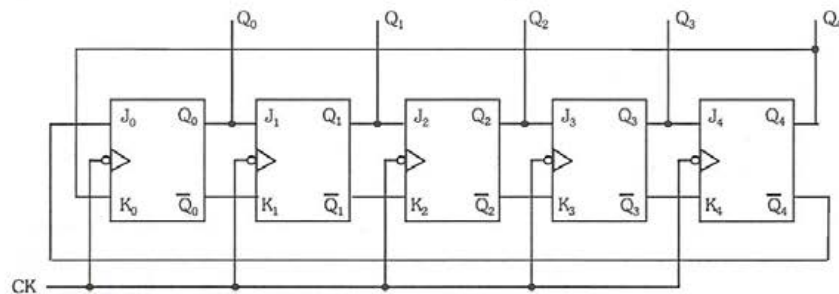



Figura 6.67

Podemos notar que, sendo o estado inicial 0, nas entradas J_0 e K_0 teremos 1 e 0 respectivamente. Logo após o 1º pulso de clock, o contador irá apresentar o seguinte estado:

Q_0	Q_1	Q_2	Q_3	Q_4
1	0	0	0	0

Esta realimentação ($J_0 = \overline{Q}_4$ e $K_0 = Q_4$) irá fazer com que o contador execute a seqüência do código Johnson sucessivamente, em função dos pulsos de clock, conforme a tabela:

Clock	Q_4	Q_3	Q_2	Q_1	Q_0
1º	0	0	0	0	0
2º	0	0	0	0	1
3º	0	0	0	1	1
4º	0	0	1	1	1
5º	0	1	1	1	1
6º	1	1	1	1	1
7º	1	1	1	1	0
8º	1	1	1	0	0
9º	1	1	0	0	0
10º	1	0	0	0	0
	0	0	0	0	0

Tabela 6.27

Após o 5º pulso de clock, notamos que a saída Q_4 torna-se 1, logo, $\overline{Q}_4$ = 0. Isso fará com que as entradas J_0 e K_0 fiquem iguais a 0 e 1 respectivamente, gerando a continuação da seqüência vista na tabela 6.27.

Para forçar o contador a iniciar no estado 0, podemos, logo de início, zerar o contador, ou seja, aplicar uma descida de pulso nas entradas clear de todos os flip-flops do circuito, podendo estas ser interligadas.

6.4.2.6 Exercícios Resolvidos

1 - Projete um contador síncrono de 3 bits efetuar a contagem crescente (X = 0 ⇨ 0 a 7_{10}), ou decrescente (X = 1 ⇨ 7_{10} a 0), através de uma variável de controle X.

Os contadores síncronos crescentes e decrescentes são casos particulares dos contadores geradores de seqüências quaisquer. A única diferença será a introdução da variável X, que quando estiver em 0, fará com que o contador efetue a contagem crescente, e quando estiver em 1, efetue a contagem decrescente. A tabela 6.28 apresenta a seqüência proposta.

	X	Q_2	Q_1	Q_0	
Contagem Crescente	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	2
	0	0	1	1	3
	0	1	0	0	4
	0	1	0	1	5
	0	1	1	0	6
	0	1	1	1	7
Contagem Decrescente	1	1	1	1	7
	1	1	1	0	6
	1	1	0	1	5
	1	1	0	0	4
	1	0	1	1	3
	1	0	1	0	2
	1	0	0	1	1
	1	0	0	0	0

Tabela 6.28

Vamos, inicialmente, estudar o comportamento das entradas JK:

X	Q_2	Q_1	Q_0	J_2	K_2	J_1	K_1	J_0	K_0
0	0	0	0	0	X	0	X	1	X
0	0	0	1	0	X	1	X	X	1
0	0	1	0	0	X	X	0	1	X
0	0	1	1	1	X	X	1	X	1
0	1	0	0	X	0	0	X	1	X
0	1	0	1	X	0	1	X	X	1
0	1	1	0	X	0	X	0	1	X
0	1	1	1	X	1	X	1	X	1
1	1	1	1	X	0	X	0	X	1
1	1	1	0	X	0	X	1	1	X
1	1	0	1	X	0	0	X	X	1
1	1	0	0	X	1	1	X	1	X
1	0	1	1	0	X	X	0	X	1
1	0	1	0	0	X	X	1	1	X
1	0	0	1	0	X	0	X	X	1
1	0	0	0	1	X	1	X	1	X

Tabela 6.29

A seguir, vamos simplificar o circuito das entradas J e K dos flip-flops, através dos diagramas de Veitch-Karnaugh:

J_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{X}\,\overline{Q}_2$	0	0	1	0
$\overline{X}\,Q_2$	X	X	X	X
$X\,Q_2$	X	X	X	X
$X\,\overline{Q}_2$	1	0	0	0

(a) $J_2 = X\overline{Q}_1\overline{Q}_0 + \overline{X}Q_1Q_0$

K_2:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{X}\,\overline{Q}_2$	X	X	X	X
$\overline{X}\,Q_2$	0	0	1	0
$X\,Q_2$	1	0	0	0
$X\,\overline{Q}_2$	X	X	X	X

(b) $K_2 = X\overline{Q}_1\overline{Q}_0 + \overline{X}Q_1Q_0$

J_1:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{X}\,\overline{Q}_2$	0	1	X	X
$\overline{X}\,Q_2$	0	1	X	X
$X\,Q_2$	1	0	X	X
$X\,\overline{Q}_2$	1	0	X	X

(c) $J_1 = X\overline{Q}_0 + \overline{X}Q_0$

$\therefore J_1 = K_1 = X \oplus Q_0$

K_1:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{X}\,\overline{Q}_2$	X	X	1	0
$\overline{X}\,Q_2$	X	X	1	0
$X\,Q_2$	X	X	0	1
$X\,\overline{Q}_2$	X	X	0	1

(d) $K_1 = X\overline{Q}_0 + \overline{X}Q_0$

J_0:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{X}\,\overline{Q}_2$	1	X	X	1
$\overline{X}\,Q_2$	1	X	X	1
$X\,Q_2$	1	X	X	1
$X\,\overline{Q}_2$	1	X	X	1

(e) $J_0 = 1$

K_0:

	$\overline{Q}_1\overline{Q}_0$	$\overline{Q}_1 Q_0$	$Q_1 Q_0$	$Q_1\overline{Q}_0$
$\overline{X}\,\overline{Q}_2$	X	1	1	X
$\overline{X}\,Q_2$	X	1	1	X
$X\,Q_2$	X	1	1	X
$X\,\overline{Q}_2$	X	1	1	X

(f) $K_0 = 1$

Figura 6.68

Vamos, agora, esquematizar o circuito de um contador crescente ou decrescente:

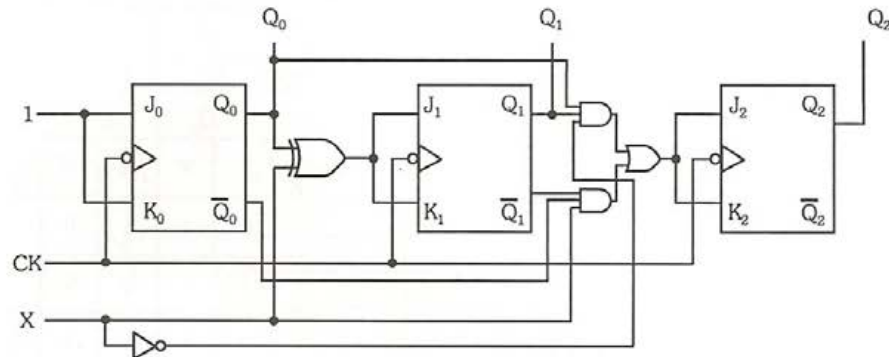



Figura 6.69

2 - Determine o diagrama de estados para o contador da figura 6.70, sabendo-se que no instante inicial os flip-flops foram "resetados".

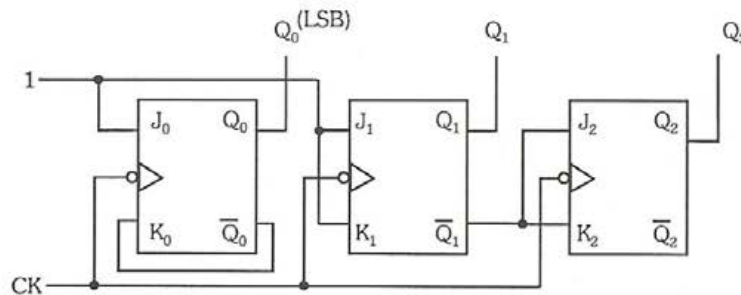



Figura 6.70

Para determinarmos o diagrama de estados, devemos obter as situações das saídas deste contador síncrono, em função das entradas J e K dos flip-flops a cada descida de clock. Assim sendo, temos:

situação inicial	⇨	$Q_0 = 0$ (CLR = 0)	$Q_1 = 0$ (CLR = 0)	$Q_2 = 0$ (CLR = 0)
entradas J e K:	⇨	$J_0 = 1, K_0 = 1$	$J_1 = 1, K_1 = 1$	$J_2 = 1, K_2 = 1$
1ª descida	⇨	$Q_0 = 1$ ($Qf = \overline{Q}a$)	$Q_1 = 1$ ($Qf = \overline{Q}a$)	$Q_2 = 1$ ($Qf = \overline{Q}a$)
entradas J e K:	⇨	$J_0 = 1, K_0 = 0$	$J_1 = 1, K_1 = 1$	$J_2 = 0, K_2 = 0$
2ª descida	⇨	$Q_0 = 1$ ($Qf = 1$)	$Q_1 = 0$ ($Qf = \overline{Q}a$)	$Q_2 = 1$ ($Qf = Qa$)
entradas J e K:	⇨	$J_0 = 1, K_0 = 0$	$J_1 = 1, K_1 = 1$	$J_2 = 1, K_2 = 1$

3ª descida	⇨ $Q_0 = 1$ (Qf =1)	$Q_1 = 1$ (Qf = $\overline{Q}$a)	$Q_2 = 0$ (Qf = $\overline{Q}$a)
entradas J e K:	⇨ $J_0 = 1, K_0 = 0$	$J_1 = 1, K_1 = 1$	$J_2 = 0, K_2 = 0$
4ª descida	⇨ $Q_0 = 1$ (Qf =1)	$Q_1 = 0$ (Qf = $\overline{Q}$a)	$Q_2 = 0$ (Qf =Qa)
entradas J e K:	⇨ $J_0 = 1, K_0 = 0$	$J_1 = 1, K_1 = 1$	$J_2 = 1, K_2 = 1$
5ª descida	⇨ $Q_0 = 1$ (Qf =1)	$Q_1 = 1$ (Qf = $\overline{Q}$a)	$Q_2 = 1$ (Qf = $\overline{Q}$a)

Notamos que após a 5ª descida de clock, o contador irá voltar às mesmas saídas obtidas após a 1ª descida, repetindo o ciclo de contagem.

A tabela 6.30 apresenta o resultado das saídas, obtidas da análise efetuada, na ordem conveniente.

Q_2	Q_1	Q_0
0	0	0
1	1	1
1	0	1
0	1	1
0	0	1

Tabela 6.30

A partir da tabela, concluímos que o contador do caso 0 irá para o 7_{10}, deste para o 5_{10}, deste para o 3_{10}, deste para o 1_{10} e deste para o 7_{10}, reiniciando o ciclo de contagem. Assim sendo, obtemos o diagrama de estados visto na figura 6.71.

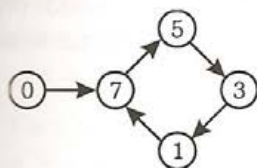



Figura 6.71

6.4.3 Contadores Utilizados em Circuitos Temporizadores

Os contadores podem ser usados em várias aplicações nos sistemas digitais. Nos itens subseqüentes, vamos destacar as aplicações em sistemas temporizadores a relógios digitais.

6.4.3.1 Contador de 0 a 59

Este é um contador muito utilizado nos tipos de circuitos mencionados, pois a cada 60 segundos deve contar 1 minuto e a cada 60 minutos deve contar 1 hora.

Podemos construir um contador de 0 a 59 de várias maneiras. A primeira é montar um contador assíncrono de 0 a n, onde n é igual a 59. O processo de obtenção deste tipo de contador foi visto no item 6.4.1.3.

A segunda maneira de utilizar dois contadores assíncronos, sendo um de 0 a 9_{10} (contador de década) e outro de 0 a 5_{10}, ligados conforme mostra a figura 6.72.

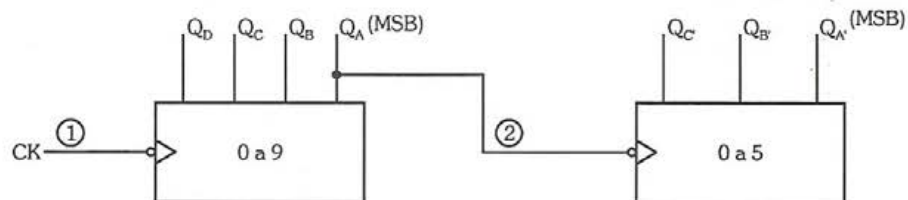



Figura 6.72

Notamos que a cada 10 pulsos na entrada 1, teremos uma descida de pulso na entrada 2, e após o pulso 60, teremos o contador novamente no estado inicial.

A terceira maneira é utilizar um contador síncrono que execute a seqüência de 0 a 59 (item 6.4.2.3), porém para levantarmos a tabela da verdade deste contador, o trabalho é exaustivo, pois precisamos utilizar 6 flip-flops.

A quarta maneira é utilizar dois contadores síncronos, sendo um de década e o outro de 0 a 5_{10}, ligados de maneira análoga ao sistema visto na figura 6.72 com contadores assíncronos.

6.4.3.2 Contador de 1 a 12

Este contador é utilizado para a contagem de horas. No caso da contagem de 1 a 12, é mais utilizado o contador síncrono, pois este permite facilmente estabelecer o início da contagem pelo projeto. Para esquematizarmos, basta que sigamos o procedimento descrito no item 6.4.2.3.

6.4.3.3 Diagrama de Blocos de um Relógio Digital

Com os elementos vistos até aqui, podemos esquematizar o diagrama de blocos de um relógio digital básico. Este é visto na figura 6.73.

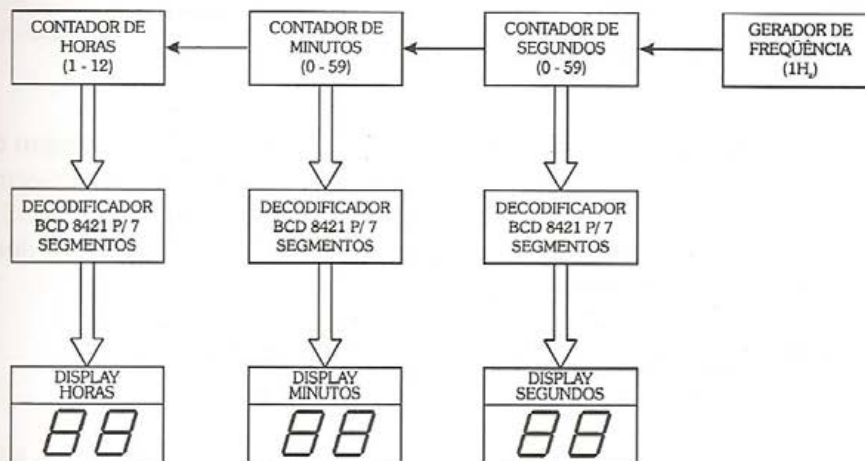



Figura 6.73

Analisando este diagrama de blocos, notamos que a cada pulso do gerador de freqüência, o contador de segundos apresenta sua contagem num display de 7 segmentos, gerando também o pulso de clock para o contador de minutos, que também apresenta contagem no display de minutos. Este contador, por sua vez, gera o pulso de clock para o contador de horas. Assim sendo, podemos ver nos displays a contagem relativa às horas, minutos e segundos do relógio.

6.4.3.4 Exercícios Resolvidos

1 - Escolha dois blocos contadores e interligue-os de maneira a formar um sistema contador de 0 a 29_{10}. O sistema deverá ter uma entrada para estabelecer o clear inicial.

Para realizar uma contagem de 0 a 29_{10}, necessitamos de um contador de 0 a 9_{10} para o dígito menos significativo e outro de 0 a 2_{10} para o mais significativo. Estes contadores podem ser assíncronos ou síncronos, sendo a ligação dos blocos de maneira assíncrona, tendo a saída do bit mais significativo (MSB) do contador de 0 a 9_{10}, ligada à entrada clock do outro de 0 a 2_{10}, para movimentá-lo na descida do 10º pulso. A figura 6.74 mostra o esquema da ligação para que tal contagem se realize, bem como, mostra a interligação das entradas clear.

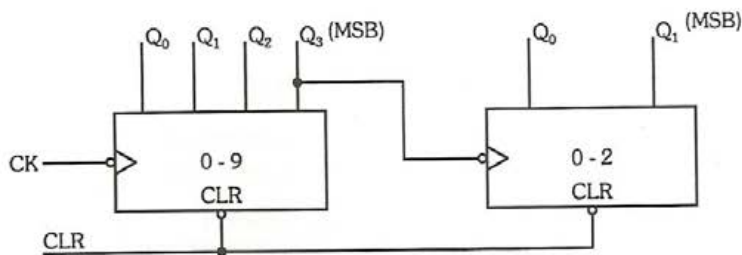



Figura 6.74

2 - Modifique o circuito do exercício anterior, para efetuar a contagem de 0 a 27_{10} (28 estados). O circuito deverá manter a entrada para estabelecimento do clear inicial.

A tabela 6.31 apresenta a contagem de 0 a 27_{10}, obtida através dos dois módulos separadamente.

Q'_1	Q'_0		Q_3	Q_2	Q_1	Q_0	
0	0	(0)	0	0	0	0	(0)
		⋮			⋮		
0	0	(0)	1	0	0	1	(9)
0	1	(1)	0	0	0	0	(0)
		⋮			⋮		
0	1	(1)	1	0	0	1	(9)
1	0	(2)	0	0	0	0	(0)
		⋮			⋮		
1	0	(2)	0	1	1	0	(6)
1	0	(2)	0	1	1	1	(7)

Tabela 6.31

Pela tabela observamos que cada ciclo completo do contador de 0 a 9_{10} incrementa uma unidade no contador de 0 a 2_{10} até o sistema chegar conjuntamente ao número 27_{10}.

Para que o circuito volte a 0, após o caso 27, ou seja, no caso 28, será necessária a colocação no clear geral, de uma porta NE com as entradas ligadas em Q_3 do contador de 0 a 9_{10} e Q_1 do de 0 a 2_{10}, pois 28 será obtido pela composição de 2 ($Q'_1 = 1$ e $Q'_0 = 0$) e 8 ($Q_3 = 1$, $Q_2 = 0$, $Q_1 = 0$ e $Q_0 = 0$).

O clear geral para uso externo pode ser mantido pela simples inserção no circuito, de uma porta E, ligada entre a NE e o clear comum aos 2 blocos, ficando o outro terminal da E, responsável pelo clear. A figura 6.75 apresenta a interligação dos blocos, com todas as modificações descritas, efetuadas.

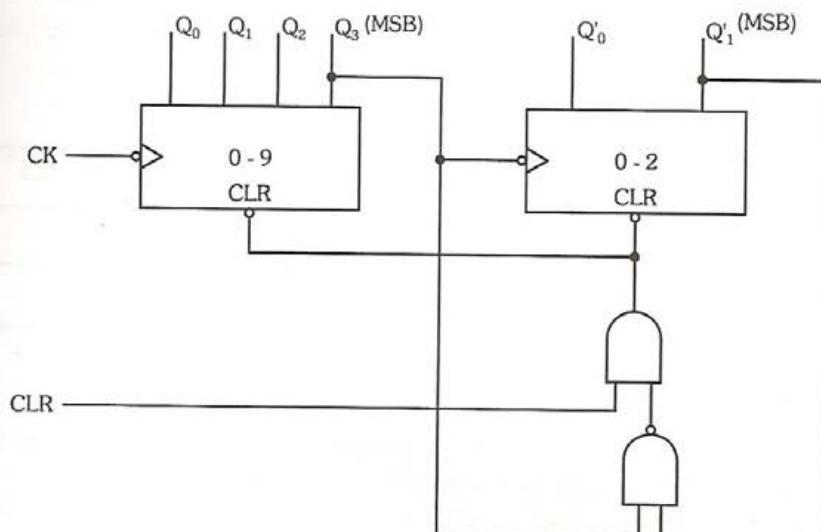



Figura 6.75

Notamos, pela figura, que aplicando 0 no terminal clear, teremos a saída da porta E em nível 0, independentemente do estado de saída do contador, levando este à situação de clear (estado 0).

6.5 Exercícios Propostos

6.5.1 Esquematize um flip-flop RS com entrada clock apenas com portas NOU. Para o circuito obtido, escreva as tabelas, mostrando a atuação de R, S e clock.

6.5.2 Em função dos sinais aplicados, determine a forma de onda da saída Q, para o flip-flop da figura 6.76.

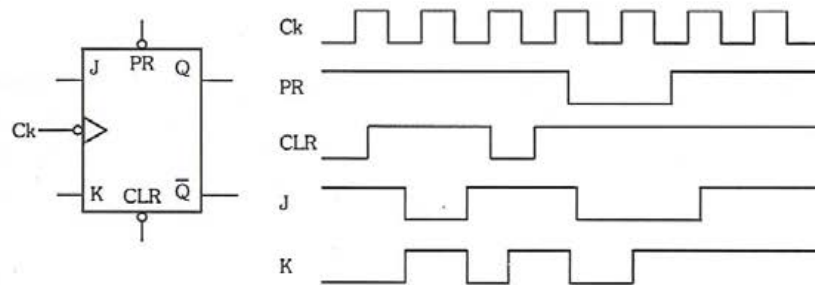



Figura 6.76

6.5.3 Idem ao anterior, para a flip-flop da figura 6.77.

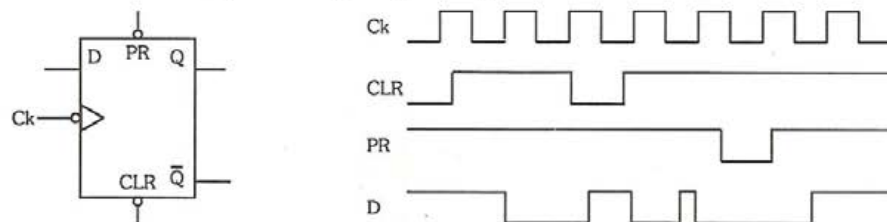



Figura 6.77

6.5.4 Esboce as formas de onda, para o registrador de deslocamento da figura 6.78, em função dos sinais aplicados, considerando a entrada enable igual a 0.

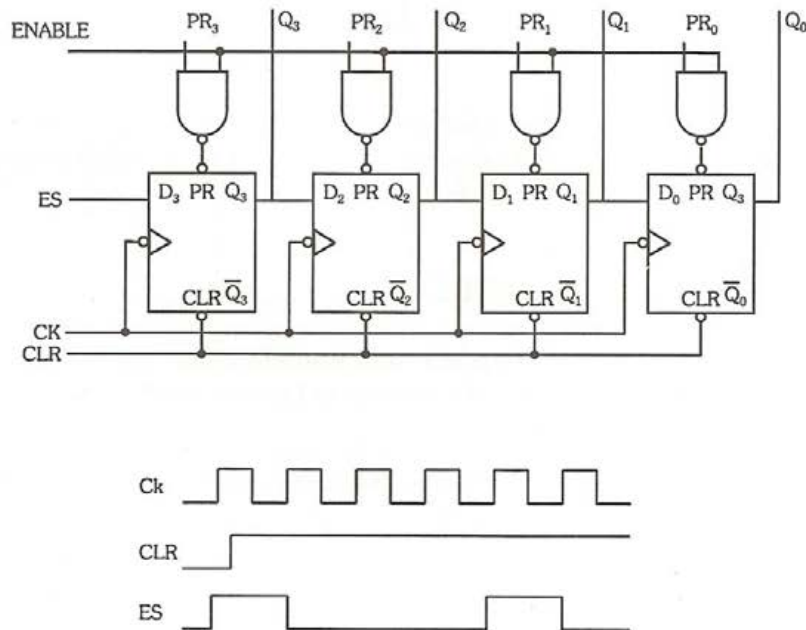



Figura 6.78

6.5.5 Determine a situação das saídas Q_3, Q_2, Q_1 e Q_0 para o circuito do exercício anterior, após 3 descidas de clock, sabendo-se que $PR_3 = 1$, $PR_2 = 0$, $PR_1 = 0$, $PR_0 = 0$ e $ES = 0$, que inicialmente houve a passagem do clear de 0 para 1, que o enable passou de 0 para 1 e logo após de 1 para 0.

6.5.6 No exercício anterior, o que aconteceria se ligássemos a saída Q_0 à entrada ES e, logo após, aplicássemos à entrada clock sucessivas descidas de pulsos?

6.5.7 A figura 6.79 mostra a situação de saída de um registrador de deslocamento de 6 bits, configurado para efetuar deslocamento à esquerda. Determine a nova situação de saída, no caso do pulso de clock aplicado ao sistema descer 2 vezes.

Q_5	Q_4	Q_3	Q_2	Q_1	Q_0
0	0	0	1	1	0

Figura 6.79

6.5.8 No exercício anterior, o que aconteceu numericamente com o resultado, após a aplicação dos pulsos?

6.5.9 Elabore um contador assíncrono de 0 a 8_{10}.

6.5.10 Altere o circuito obtido no exercício anterior, colocando uma entrada clear no contador para utilização externa.

6.5.11 Desenhe um contador assíncrono de 1 a 12_{10}. O circuito deve possuir uma entrada para estabelecer o caso inicial, através do nível 0 aplicado.

6.5.12 Esquematize um contador para trabalhar como divisor de freqüência por 50.

6.5.13 Elabore um contador assíncrono de 9_{10} a 0. O circuito deve possuir um terminal que, quando aterrado, estabelece o caso inicial (9_{10}).

6.5.14 Idem ao exercício anterior, para um contador de 18_{10} a 0.

6.5.15 Desenhe o circuito de um contador assíncrono de 0 a 3_{10} para operar de forma crescente/decrescente, conforme nível aplicado a uma entrada X de controle ($X = 1 =>$ crescente e $X = 0 =>$ decrescente).

6.5.16 Elabore o circuito de um contador síncrono que execute a seqüência mostrada no diagrama da figura 6.80. Considere os casos não pertencentes ao diagrama, como irrelevantes.

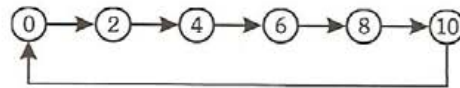



Figura 6.80

6.5.17 Projete um contador síncrono para gera a seqüência do código Excesso 3, conforme diagrama de estados visto na figura 6.81.

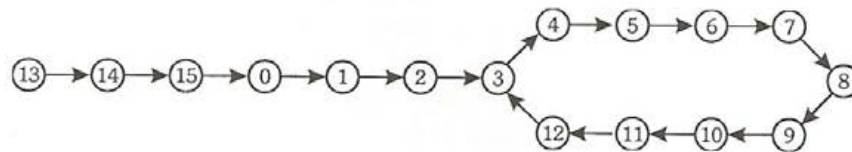



Figura 6.81

6.5.18 Obtenha as expressões simplificadas dos flip-flops de um contador síncrono para gerar a seqüência de 9_{10} a 0.

6.5.19 Determine o diagrama de estados do contador visto na figura 6.82. Considere que inicialmente a entrada clear foi acionada.

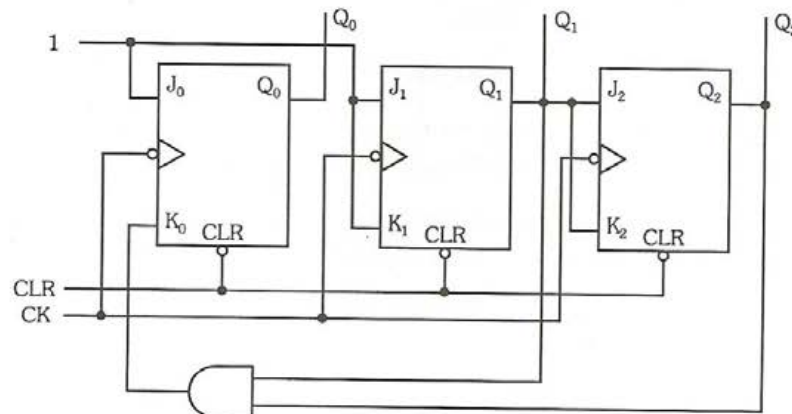
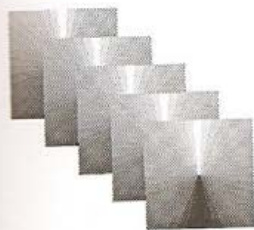



Figura 6.82

6.5.20 Escolha 2 blocos contadores e interligue-os de maneira a formar um sistema contador de 0 a 54_{10}. Desenhe o esquema de ligação, colocando uma entrada clear para utilização externa.

CAPÍTULO 7

Conversores Digital-Analógicos e Análogo-Digitais

7.1 Introdução

Vamos, neste capítulo, tratar dos Conversores Digital-Analógicos e Análogo-Digitais. Para iniciarmos este estudo, vamos, primeiramente, estudar o significado dos termos analógico e digital.

Entende-se por analógica toda variação contínua de uma variável. Todas as grandezas físicas (velocidade, pressão, temperatura, corrente elétrica, tensão, resistência, etc) variam de forma analógica, isto é, para se atingir um valor desejado de uma grandeza qualquer, é necessário que esta passe por todos os valores intermediários de forma contínua.

Qualquer variação existente pode ser observada através de um gráfico, onde se relacionam a grandeza que varia, o tempo ou outra referência física. O gráfico da figura 7.1 mostra, a título de exemplo, uma variação contínua ou analógica genérica.

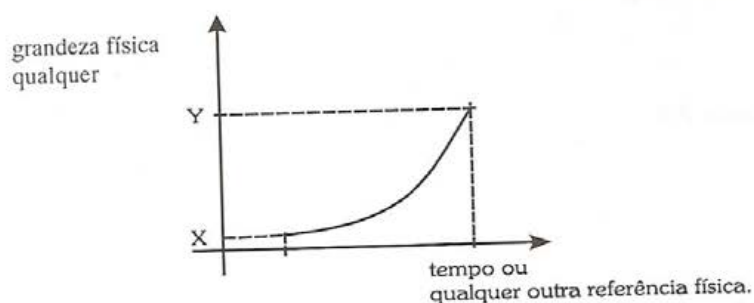



Figura 7.1

Em resumo, uma variável analógica pode assumir todos os valores dentro de sua faixa de atuação.

Entende-se por digital, toda variação discreta, ou seja, a passagem de um valor a outro se dá por saltos. Como exemplo, vamos observar na figura 7.2, o gráfico de uma variação digital.

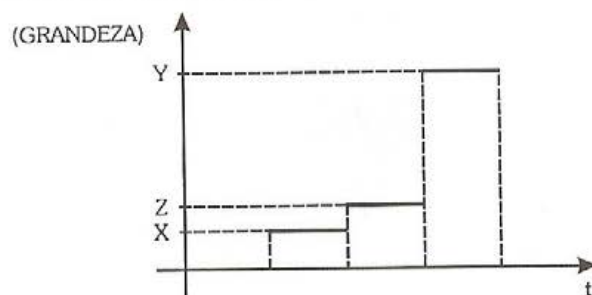



Figura 7.2

Uma conclusão imediata que podemos tirar, comparando a variação analógica com a digital, é que na primeira, entre um valor e outro, existem infinitos valores; já na segunda, possuímos um número finito de valores, no exemplo, na variação digital entre X e Y temos apenas três valores: X, Y e Z.

Vamos agora, para reforçar estes conceitos, analisar dois dispositivos: um de variação analógica e outro digital.

Variação Analógica: Potenciômetro

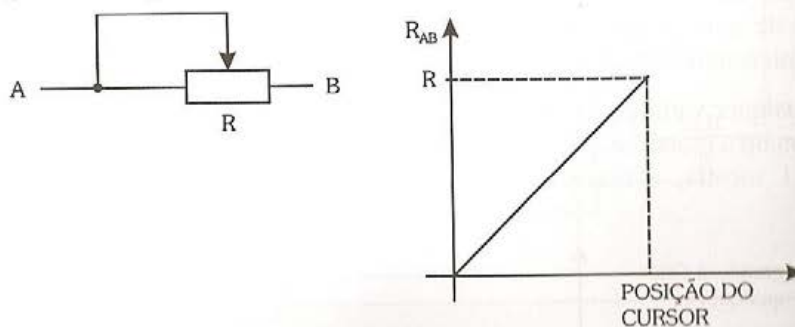



Figura 7.3

Variação Digital: Chave Seletora

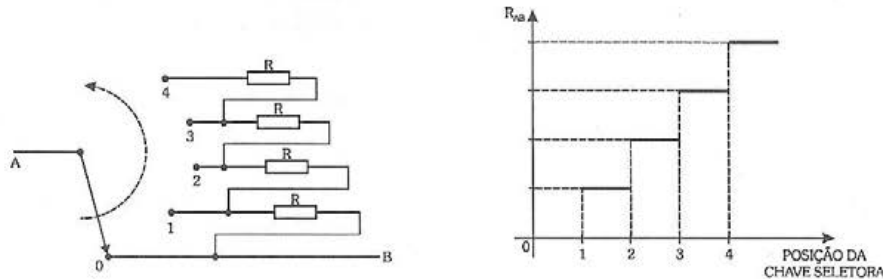



Figura 7.4

Como outros exemplos de variações digitais, vamos citar os códigos digitais, pois nestes, passamos de um estado para outro sem infinitos valores intermediários. Dentre os códigos digitais destacamos o BCD 8421, de aplicação mais comum em conversores.

Em vários casos na eletrônica digital, necessitamos converter sinais analógicos em digitais e vice-versa. Para estas aplicações, utilizamos os conversores análogo-digitais e conversores digital-analógicos, respectivamente.

Estes circuitos são muito utilizados em instrumentação digital, transmissão de informações de forma digital e em outros sistemas que, da mesma forma, relacionam variações analógicas com variações digitais.

7.2 Conversores Digital-Analógicos

Este circuito é utilizado quando necessitamos converter uma variação digital em analógica. A informação digitalizada, geralmente, é feita no código BCD 8421 e é a partir deste, que se faz a conversão para a saída analógica. Na saída analógica, teremos esta mesma informação em níveis de tensão correspondentes ao valor binário injetado na entrada. A figura 7.5 mostra a estrutura geral de um Conversor Digital-Analógico (D/A).

Figura 7.5

7.2.1 Conversor Digital-Analógico Básico

O circuito apresentado a seguir, é o mais simples que efetua a conversão digital-analógica. Trata-se de um circuito que utiliza como componentes apenas resistores.

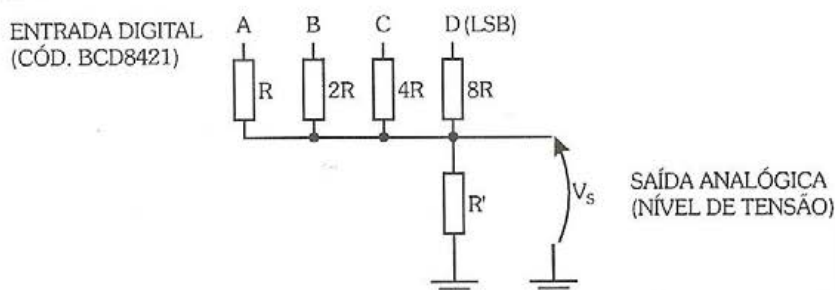



Figura 7.6

Para entendermos o funcionamento do circuito, devemos lembrar que o nível 0 de tensão corresponde a 0V, ou seja, equivale a ligarmos o ponto ao terra; e o nível 1 de tensão correspondente a uma tensão predeterminada, geralmente igual a Vcc. Outra consideração que devemos observar é que R', que é o resistor no qual iremos ter a tensão de saída, terá que ser muito menor que R para que não influa no circuito.

Se tivermos nível 1 em A e 0 nas demais entradas (1000_2), a tensão de R' será:

$$Vs = \frac{Vcc \cdot R'}{R + R'}, \text{ como } R' << R\text{: } Vs = \frac{Vcc \cdot R'}{R}$$

Se tivermos nível 1 em B e 0 nas demais entradas (0100_2), a tensão R' será:

$$Vs = \frac{Vcc \cdot R'}{2 \cdot R}$$

Podemos observar que neste último caso, o valor da tensão Vs será a metade do caso anterior.

Continuando, se tivermos nível 1 na entrada C e 0 nas demais entradas (0010_2), a tensão de saída será:

$$Vs = \frac{Vcc \cdot R'}{4 \cdot R}$$

Por último, se tivermos nível 1 na entrada D e 0 nas demais entradas (0001_2), a tensão de saída será:

$$Vs = \frac{Vcc \cdot R'}{8 \cdot R}$$

Se considerarmos esta última saída igual a 1V, teremos que as anteriores serão proporcionalmente 2, 4 e 8 V.

Se tivermos, por exemplo, as entradas A e C em 1 e as demais em 0 ($1010_2 = 10_{10}$), teremos a seguinte tensão de saída:

$$Vs = \frac{Vcc \cdot R'}{R} + \frac{Vcc \cdot R'}{4 \cdot R} = Vs = \frac{Vcc \cdot R'}{R} \cdot \left(1 + \frac{1}{4}\right) =$$

$$= \frac{Vcc \cdot R'}{R} \cdot \frac{5}{4} \therefore Vs = \frac{Vcc \cdot R' \cdot 5}{4 \cdot R}$$

Se compararmos esta tensão de saída com a tensão que foi considerada como referência, veremos que é dez vezes maior:

$$\frac{\dfrac{Vcc \cdot R' \cdot 5}{4 \cdot R}}{\dfrac{Vcc \cdot R'}{8 \cdot R}} = 10$$

Dando-se valores adequados aos resistores e a Vcc, teremos na saída uma tensão proporcional à entrada injetada.

Para fixarmos melhor o funcionamento do circuito básico, vamos dar valores aos elementos do circuito e verificar, a título de exemplo, algumas das possíveis conversões:

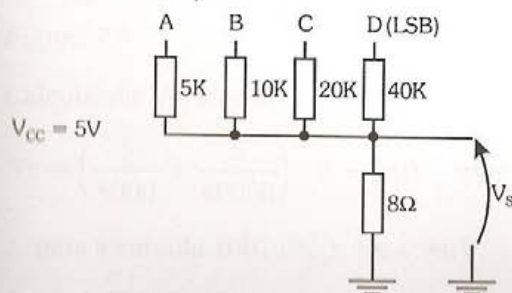



Figura 7.7

Para o valor de R, adotamos 5KΩ e para R' 8Ω, fazendo com que, na saída, tenhamos um valor numericamente proporcional à entrada.

Para fixarmos o funcionamento, vamos exemplificar algumas conversões:

Exemplo 1:

Entrada:

A	B	C	D
0	0	0	0

$\leftrightarrow 0_{10}$

Temos neste caso, as tensões de entrada iguais a 0 e obviamente uma tensão de saída igual a 0V.

$\therefore$ Vs = 0V

Exemplo 2:

Entrada:

A	B	C	D
0	1	0	1

$\leftrightarrow 5_{10}$

Neste caso, temos:

0V na entrada A

5V na entrada B (Vcc = 5V)

0V na entrada C

5V na entrada D (Vcc = 5V)

O circuito com estes níveis de entrada é visto na figura 7.8.

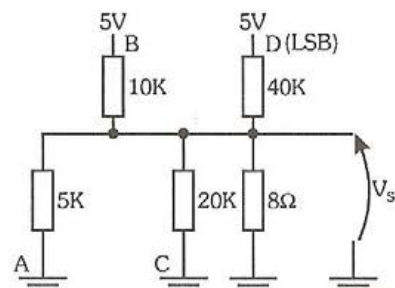



Figura 7.8

Calculando Vs, temos:

$$Vs = 8.\left(\frac{5}{10000} + \frac{5}{40000}\right) = 5mV$$

$\therefore$ para a entrada 0101 (5_{10}): Vs = 5mV.

No exemplo, podemos notar a proporcionalidade entre o valor digital da entrada e o valor analógico de tensão de saída.

Exemplo 3:

Entrada:

A	B	C	D
1	0	0	1

$\leftrightarrow 9_{10}$

Neste caso, temos:

5V na entrada A

0V na entrada B

0V na entrada C

5V na entrada D

O circuito com os níveis é visto na figura 7.9.

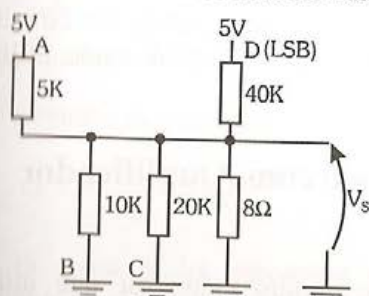



Figura 7.9

Calculando Vs, temos:

$$Vs = \left(\frac{5}{5000} + \frac{5}{40000}\right) . 8 = 9mV$$

$\therefore$ para a entrada 1001 (9_{10}): Vs = 9mV.

A tabela 7.1 mostra a conversão de todos os casos do código BCD 8421, através deste circuito básico:

Entrada digital				Saída Analógica
A	B	C	D	V (mV)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Tabela 7.1

Se considerarmos na entrada os casos superiores a 9_{10}, pertencentes ao código BCD8421, obteremos, da mesma forma, os níveis correspondentes de sinais analógicos.

O circuito básico, apesar de apresentar um funcionamento correto, possui uma característica desvantajosa que é a de apresentar um baixo valor de tensão de saída. Para resolvermos esse problema, utilizaremos um circuito um pouco mais sofisticado, fazendo uma amplificação do sinal de saída, utilizando um amplificador operacional.

7.2.2 Conversor Digital-Analógico com Amplificador Operacional

Antes de iniciarmos o estudo do circuito conversor que utiliza o amplificador operacional, vamos fazer algumas considerações básicas sobre este último.

Características principais do amplificador operacional:

1) Alta impedância de entrada.
2) Baixa impedância de saída.
3) Tensão de saída igual a 0 quando as entradas 1 e 2 tiverem a mesma tensão.

A simbologia utilizada para este bloco é vista na figura 7.10.

Entrada 1: inversora

Entrada 2: não inversora

+Vcc

① -

② +

S (saída)

-Vcc

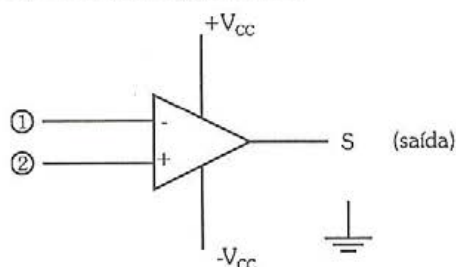



Figura 7.10

A figura 7.11 mostra a montagem de um amplificador inversor de ganho estabilizado com utilização do amplificador operacional:

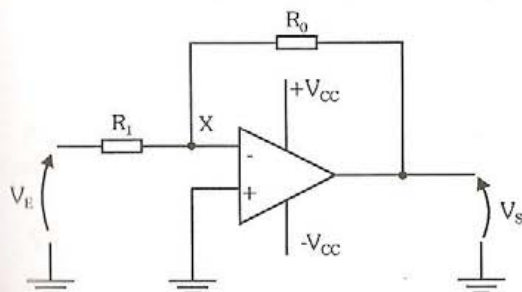



Figura 7.11

O ganho do amplificador apresentado será:

$$G = -\frac{Vs}{Ve} = -\frac{R_0}{R_1}$$

O ponto X irá apresentar um baixo potencial, pois o amplificador operacional apresenta como característica básica um elevado ganho, vem daí este ponto ser conhecido como terra virtual, pois esse baixo potencial será praticamente o mesmo da entrada não inversora que está ligada à massa.

Outra característica importante é a saturação da tensão de saída, que na realidade, é limitada pela tensão de alimentação do amplificador operacional, fato devido à saturação dos circuitos internos.

Uma importante utilização do amplificador operacional é a de **circuito comparador**, o que executa a comparação de duas tensões, aplicadas às entradas inversoras e não inversoras. Quando a tensão de entrada inversora for maior que a outra, o operacional terá na saída a tensão de -Vcc, pois ocorrerá a

saturação. No caso contrário, a saída estará em +Vcc. Quando as tensões forem estritamente iguais, o operacional apresentará a saída 0. Qualquer diferença fará com que o operacional sature em +Vcc ou -Vcc, pois por menor que seja, será amplificada por um ganho elevado, fazendo assim com que a saída entre em saturação. Esta aplicação do amplificador operacional será utilizada no circuito conversor análogo-digital (item 7.3).

A montagem de um somador de tensões, utilizando o amplificador operacional, é vista na figura 7.12.

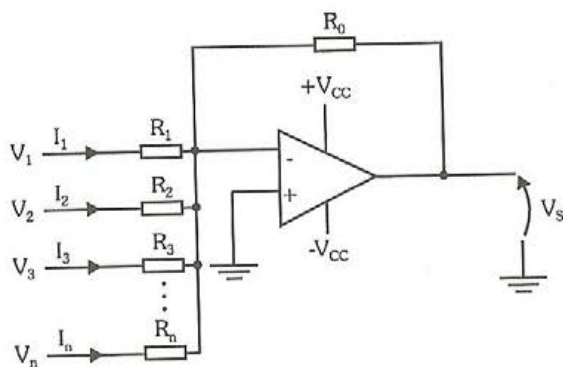



Figura 7.12

Este circuito irá apresentar a seguinte tensão de saída:

$$Vs = -\left(\frac{R_0}{R_1} \cdot V1 + \frac{R_0}{R_2} \cdot V2 + \frac{R_0}{R_3} \cdot V3 + \ldots + \frac{R_0}{R_n} \cdot Vn\right)$$

Esta expressão representa uma soma ponderada das tensões.

Após essa breve apresentação do amplificador operacional, podemos mostrar o circuito de um conversor digital-analógico com a utilização do mesmo. Este circuito nada mais é que uma aplicação do circuito somador ponderado de tensões:

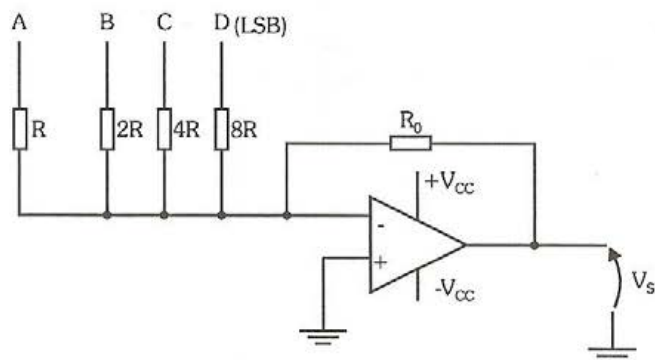



Figura 7.13

A tensão Vs é dada por:

$$Vs = -\frac{R_0}{R} \cdot \left(\frac{V_A}{1} + \frac{V_B}{2} + \frac{V_C}{4} + \frac{V_D}{8}\right)$$

As tensões V_A, V_B, V_C e V_D poderão assumir apenas dois valores: nível 1 de tensão e nível 0 de tensão, logo, podemos escrever:

$$Vs = -\frac{V \cdot R_0}{R} \cdot \left(\frac{A}{1} + \frac{B}{2} + \frac{C}{4} + \frac{D}{8}\right)$$

onde: V é a tensão de nível 1, e A, B, C e D são os bits do código BCD 8421.

Como se pode observar na expressão, a saída analógica Vs será proporcional à entrada digital, que é efetuada através do código BCD 8421.

Para mostrarmos o funcionamento do circuito, vamos elaborar alguns exemplos numéricos de conversão. Usaremos neste caso, Vcc = 16V, $R_0 = R = 5K$, para que na saída tenhamos um valor numericamente proporcional à entrada. Adotaremos, também, como nível 1 uma tensão igual a 8V.

O circuito, com os valores, é visto na figura 7.14.

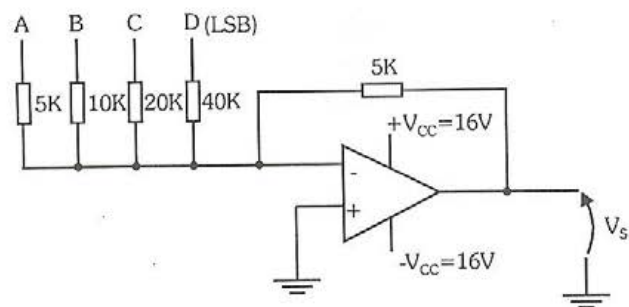



Figura 7.14

Exemplo 1:

Entrada:

A	B	C	D
0	0	1	1

$\leftrightarrow 3_{10}$

Neste caso, temos: 0V na entrada A
0V na entrada B
8V na entrada C
8V na entrada D

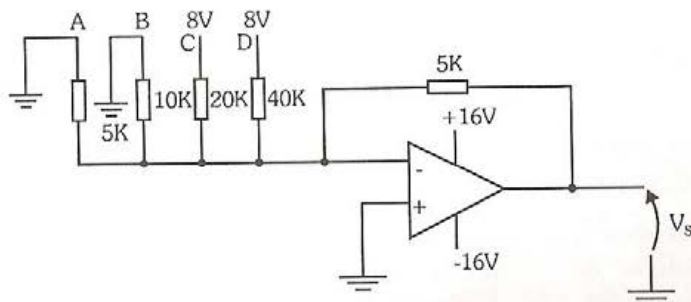



Figura 7.15

$$Vs = -\frac{8.5K}{5K}\left(\frac{1}{4}+\frac{1}{8}\right) \rightarrow Vs = -3V$$

Exemplo 2:

Entrada:

A	B	C	D
0	1	1	1

$\leftrightarrow 7_{10}$

Neste caso, temos: 0V na entrada A
8V na entrada B
8V na entrada C
8V na entrada D

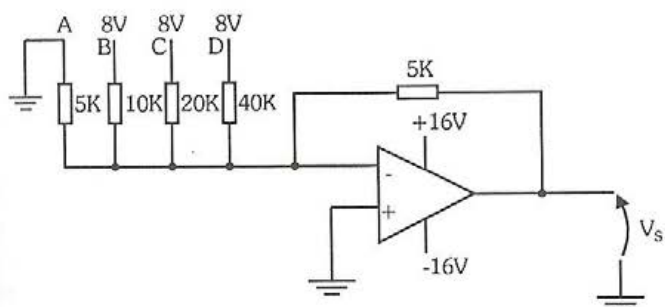



Figura 7.16

$$Vs = -\frac{8.5K}{5K} \cdot \left(\frac{1}{2} + \frac{1}{4} + \frac{1}{8}\right) \rightarrow Vs = -7V$$

Podemos notar que, com a utilização do operacional, elevamos o nível de tensão de saída de mV (mili-Volts) para V (Volts).

O quadro de conversões é visto na tabela 7.2.

Entrada Digital				Saída Analógica
A	B	C	D	V (V)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Tabela 7.2

Podemos notar, ainda, que embora o código de entrada (BCD8421) seja definido até 9_{10}, se aplicarmos os outros casos remanescentes do sistema binário comum, teremos da mesma forma a saída convertida para o correspondente nível analógico.

7.2.3 Conversor Digital-Analógico com Chave Seletora Digital

Podemos construir um circuito conversor digital-analógico com chave seletora digital na entrada. Esse circuito é praticamente análogo ao anterior, somente com a diferença de possuir em sua entrada a mencionada chave. Esta chave seletora nada mais é que um conjunto de portas E, que possuem um terminal de entrada permanente, ligado em nível 1, e outro ligado à entrada propriamente dita. A finalidade desta chave é a de isolar a impedância de saída do circuito que será ligado à entrada, fornecendo, portanto, um nível de tensão de entrada, digital bem-definido.

Seu circuito básico é visto na figura 7.17.

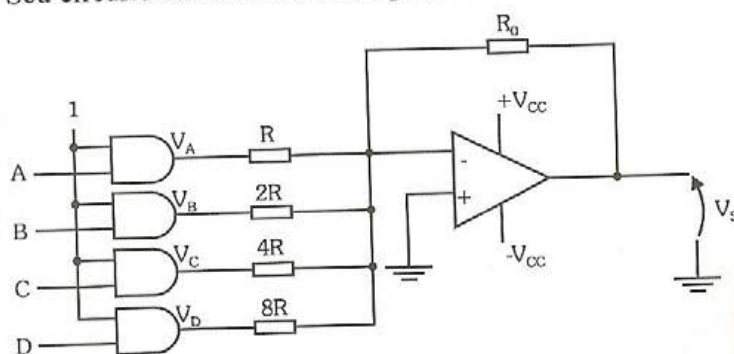



Figura 7.17

A tensão de saída terá a mesma expressão que a do circuito anterior:

$$Vs = -\frac{V.R_0}{R}.\left(A + \frac{B}{2} + \frac{C}{4} + \frac{D}{8}\right)$$

Analisando cada porta, veremos que sua saída apresentará nível 1 quando a entrada for 1, e 0 quando a entrada for 0, sendo um nível fixo e bem-definido de tensão. Os exemplos de conversão serão análogos aos do circuito anterior, visto que a configuração básica da montagem não foi alterada.

7.2.4 Conversor Digital-Analógico utilizando Rede R-2R

O circuito que estudaremos a seguir, fará a conversão digital-analógica, com a vantagem de utilizar somente resistores como componentes. O processo de conversão será explicado juntamente com o funcionamento do circuito.

O conversor Digital-Analógico utilizando rede R - 2R é visto na figura 7.18.

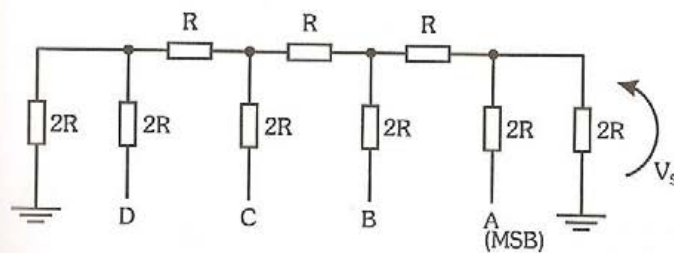



Figura 7.18

Sendo A o bit mais significativo, vamos aplicar nível 1 de tensão em A e 0 nas outras entradas. O circuito, nesta situação, é visto na figura 7.19.

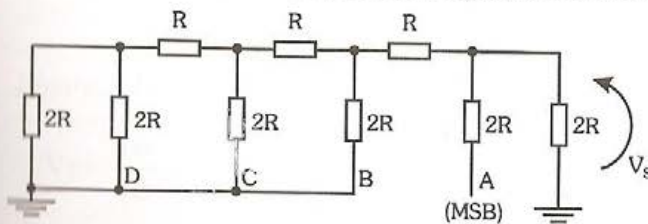



Figura 7.19

Efetuando as associações dos resistores, encontramos o circuito simplificado visto na figura 7.20.

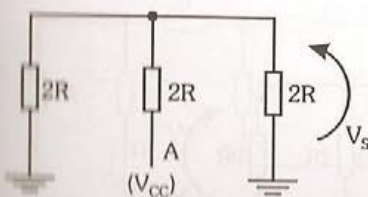



Figura 7.20

Através do divisor de tensão obtido, determinamos Vs:

$$Vs = \frac{Vcc.R}{2R + R} = \frac{Vcc}{3}$$

Vamos aplicar, agora, na entrada B, nível 1 de tensão e nas outras nível 0. O circuito, nesta situação, é visto na figura 7.21.

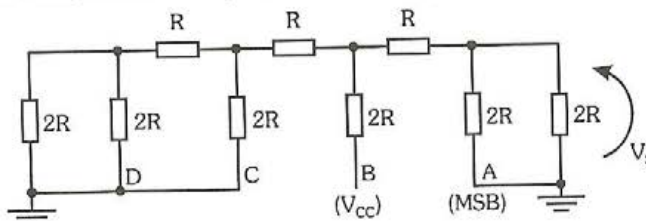



Figura 7.21

Simplificando, temos:

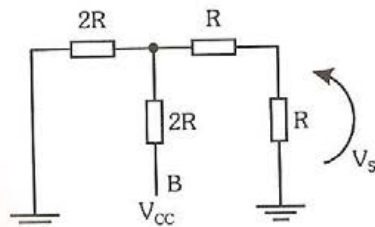



Figura 7.22

Calculando a tensão de saída, temos:

$$Vs = \frac{\frac{Vcc.R}{2R + R}}{2} = \frac{Vcc}{6}$$

Vamos aplicar agora, na entrada C, nível 1 de tensão e nas outras nível 0. O circuito, nesta situação, é visto na figura 7.23.

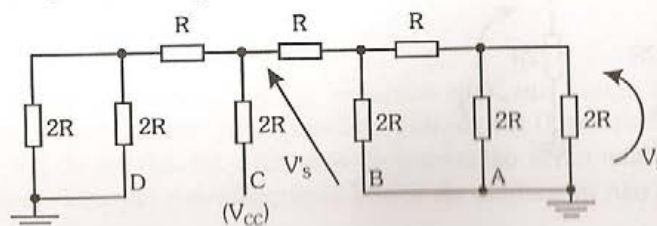



Figura 7.23

Da mesma forma, simplificando, temos:

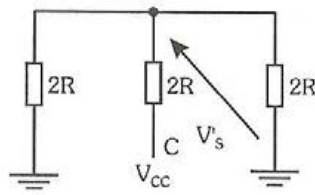



Figura 7.24

Com o intuito de calcular Vs, vamos determinar Vs':

$$Vs' = \frac{Vcc}{3}$$

A partir de Vs', obtemos Vs:

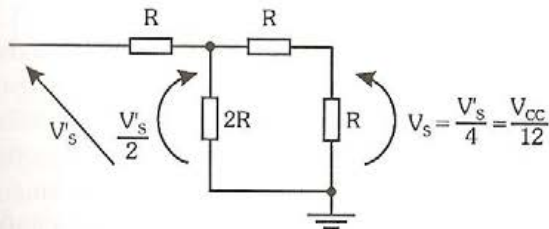



Figura 7.25

$$\therefore Vs = \frac{Vcc}{12}$$

Vamos, por último, aplicar na entrada D nível 1 de tensão, e nas outras nível 0.

Nesta situação, temos:

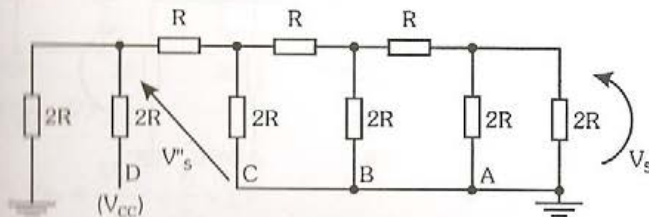



Figura 7.26

Calculando Vs", temos

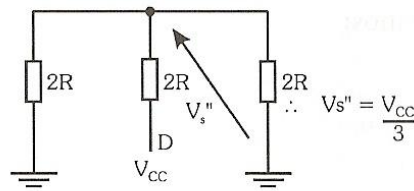



Figura 7.27

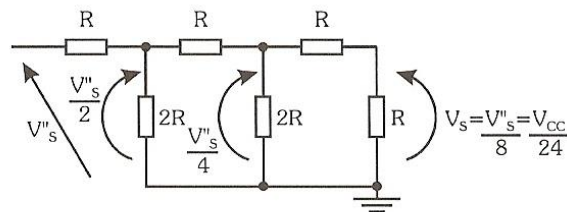



Figura 7.28

Após termos analisado cada entrada, podemos notar que para todas elas possuímos uma impedância igual a 3R, que é um fator que ajuda a manter o potencial de entrada constante. A tensão de saída, quando possuímos somente a entrada do bit mais significativo, é igual a Vcc/3 e para o bit menos significativo, a saída será 1/8 desse nível (Vcc/24). Se entrarmos com o código binário nas entradas ABCD, sendo A a entrada do bit mais significativo, teremos a tensão Vs como uma saída analógica proporcional à entrada digital. Nos casos onde temos nível 1 em mais de uma entrada, na saída aparecerá a soma ponderada das tensões, o que pode ser facilmente verificado pelo Teorema da Superposição.

Para compreendermos melhor o funcionamento do circuito, vamos estudar alguns exemplos numéricos. A figura 7.29 apresenta o mesmo circuito com valores de resistores e Vcc adotados.

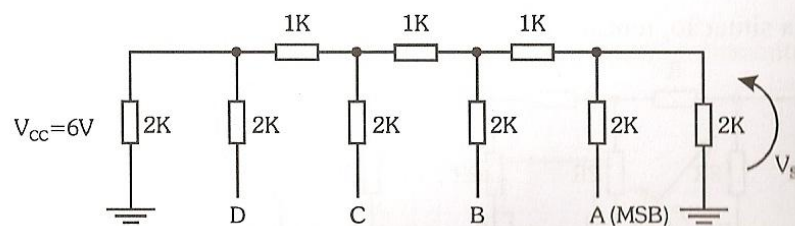



Figura 7.29

Exemplo 1: Situação de entrada:

A	B	C	D
1	1	0	0

Vamos calcular a tensão de saída para este caso:

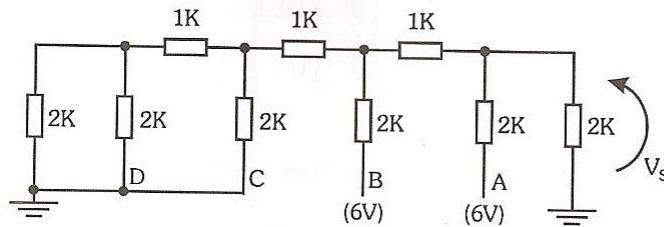



Figura 7.30

A tensão Vs poderá ser calculada, utilizando-se o Teorema da Superposição, ou seja, considerando uma fonte de cada vez:

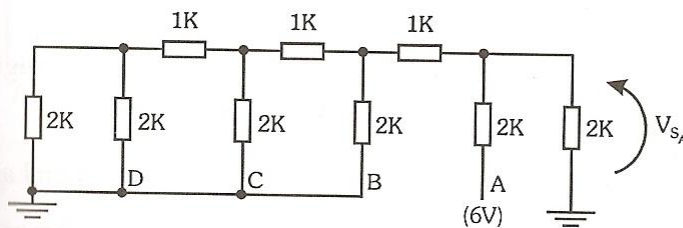



Figura 7.31

Assim sendo, temos:

$$V_{S_A} = \frac{Vcc}{3} = \frac{6}{3} = 2V$$

Considerando a outra fonte, temos:

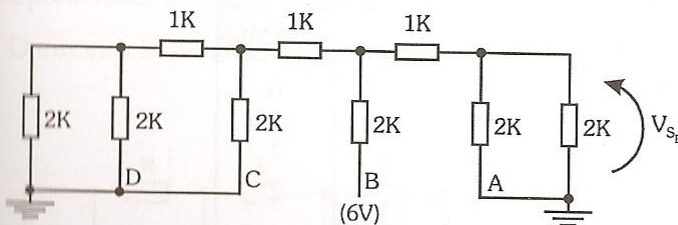



Figura 7.32

$$\therefore V_{S_B} = \frac{Vcc}{6} = \frac{6}{6} = 1V$$

Pelo teorema da superposição, podemos escrever:

$$V_S = V_{S_A} + V_{S_B} = 2 + 1 = 3V$$

Temos então que para uma entrada digital igual a 1100_2 (12_{10}), temos uma saída analógica de 3V.

Exemplo 2: Entrada:

A	B	C	D
1	0	0	0

Esta é similar à que já foi calculada no caso anterior (V_{SA}), apresentando uma tensão de saída:

$$Vs = \frac{Vcc}{3} = \frac{6}{3} = 2v$$

Logo, para uma entrada digital igual a 1000_2 (8_{10}), temos uma saída analógica de 2V.

Podemos notar que a saída não é numericamente igual ao valor digital de entrada, porém é diretamente proporcional a esse valor.

$$\begin{matrix}\text{entrada:} \\ \text{saída:}\end{matrix} \quad \frac{12_{10}}{3V} = \frac{8_{10}}{2V} = 4 \rightarrow \quad \text{fator de proporcionalidade é igual a 4.}$$

Se adotássemos um valor de nível 1 igual a 24V, o valor de saída seria numericamente igual à entrada. Na prática, porém, utiliza-se como nível 1 tensões menores, como por exemplo 5V.

No exemplo, com nível 1 igual a 6V, temos a seguinte tabela de conversão:

Entrada Digital				Saída Analógica	X4
A	B	C	D	V (V)	V (V)
0	0	0	0	0	0
0	0	0	1	0,25	1
0	0	1	0	0,50	2
0	0	1	1	0,75	3
0	1	0	0	1,00	4
0	1	0	1	1,25	5
0	1	1	0	1,50	6
0	1	1	1	1,75	7

Tabela 7.3 (parte)

Entrada Digital				Saída Analógica	X4
1	0	0	0	2,00	8
1	0	0	1	2,25	9
1	0	1	0	2,50	10
1	0	1	1	2,75	11
1	1	0	0	3,00	12
1	1	0	1	3,25	13
1	1	1	0	3,50	14
1	1	1	1	3,75	15

Tabela 7.3

7.2.5 Conversor Digital - Analógico com Rede R-2R utilizando Amplificador Operacional

O amplificador operacional é utilizado neste circuito com duas finalidades. A primeira é a de oferecer uma tensão de saída com fator de proporcionalidade qualquer, independendo da tensão fixada para nível 1, bastando para isso, modificarmos o ganho através da relação de resistências. A outra finalidade é o melhor acoplamento do conversor com outros circuitos, pois o operacional isola a impedância da rede R -2R com a carga.

O circuito básico é visto na figura 7.33.

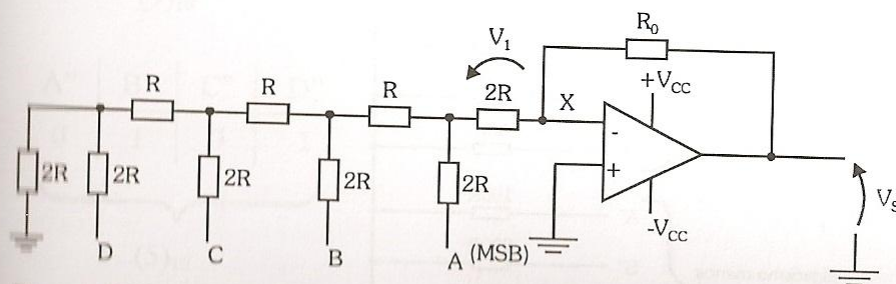



Figura 7.33

Lembrando que o ponto X pode ser considerado como sendo um ponto de terra, podemos concluir que Vs será:

$$Vs = -V_1 . \frac{R_0}{2R}$$

V_1 pode ser calculado como é mostrado no item anterior e, ainda, o ganho do operacional pode ser ajustado ao valor necessário no projeto.

7.2.6 Conversor Digital-Analógico para mais Algarismos

Podemos ter um número decimal de mais de um algarismo representado no código BCD 8421. Isto se faz, representando algarismo por algarismo através do código. Como exemplo, o número 384_{10} pode ser representado da seguinte forma:

3				8				4			
0	0	1	1	1	0	0	0	0	1	0	0
A	B	C	D	A'	B'	C'	D'	A"	B"	C"	D"

Para convertermos um número decimal de mais de um algarismo, utilizamos os circuitos básicos ampliados para recebermos outros algarismos. O circuito, para converter números com 3 algarismos, é visto na figura 7.34.

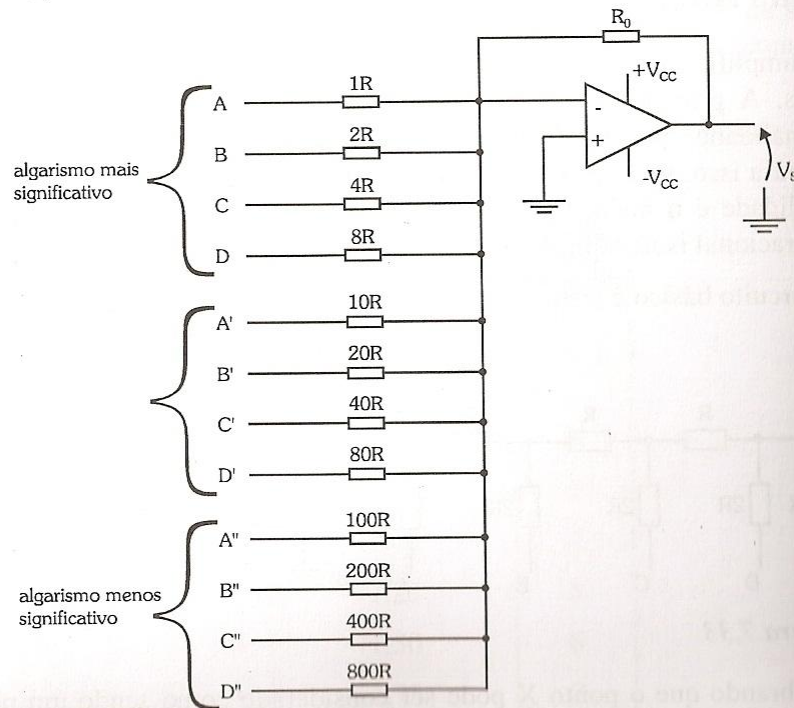



Figura 7.34

A entrada dos 4 bits que representarão o algarismo mas significativo é feita através de A, B, C e D, seguindo-se de A', B', C', D', A", B", C", D" e assim sucessivamente de acordo com a significância dos algarismos.

A tensão analógica da saída Vs terá a seguinte expressão:

$$Vs = -\frac{R_0}{R}\cdot\left[\left(\frac{V_A}{1}+\frac{V_B}{2}+\frac{V_C}{4}+\frac{V_D}{8}\right)+\left(\frac{V_{A'}}{10}+\frac{V_{B'}}{20}+\frac{V_{C'}}{40}+\frac{V_{D'}}{80}\right)\right.$$

$$\left.+\left(\frac{V_{A''}}{100}+\frac{V_{B''}}{200}+\frac{V_{C''}}{400}+\frac{V_{D''}}{800}\right)\right]$$

Para compreendermos este circuito, vamos realizar um exemplo numérico: 495_{10}. Assim sendo, temos:

Entradas:

A	B	C	D
0	1	0	0

$(4)_{10}$

A'	B'	C'	D'
1	0	0	1

$(9)_{10}$

A"	B"	C"	D"
0	1	0	1

$(5)_{10}$

Vamos aplicar cada algarismo à entrada correspondente, conforme o circuito da figura 7.35, onde foram adotados os valores dos resistores, do Vcc e do nível 1.

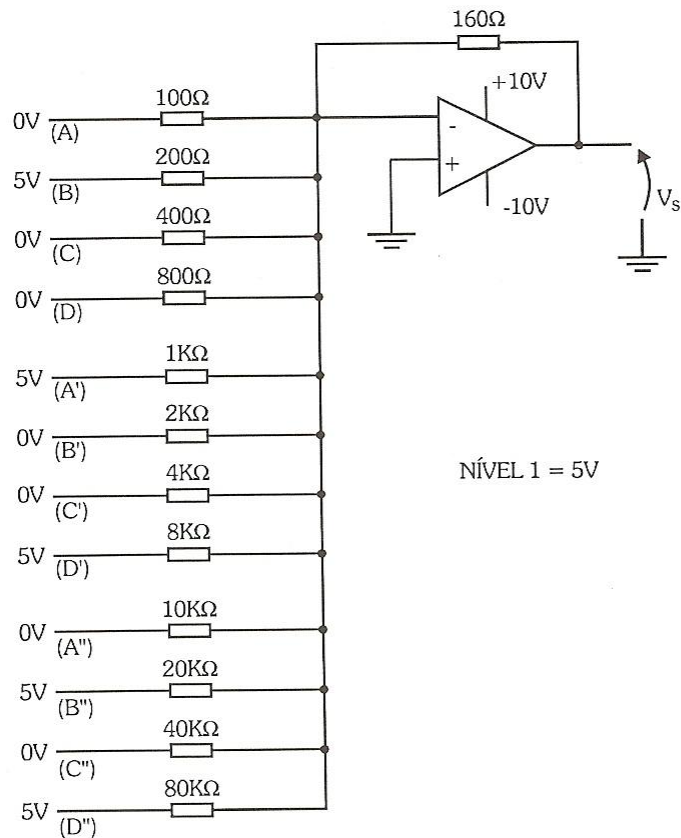



Figura 7.35

Utilizando a fórmula de Vs, podemos escrever:

$$Vs = -5.\frac{160}{100}.\left[\left(\frac{1}{2}\right)+\left(\frac{1}{10}+\frac{1}{80}\right)+\left(\frac{1}{200}+\frac{1}{800}\right)\right]$$

Vs = - 4,95V

Podemos notar a proporcionalidade de tensão de saída com os dígitos de entrada.

Podemos também, efetuar este tipo de conversão, utilizando um circuito com redes R-2R, conforme mostra a figura 7.36.

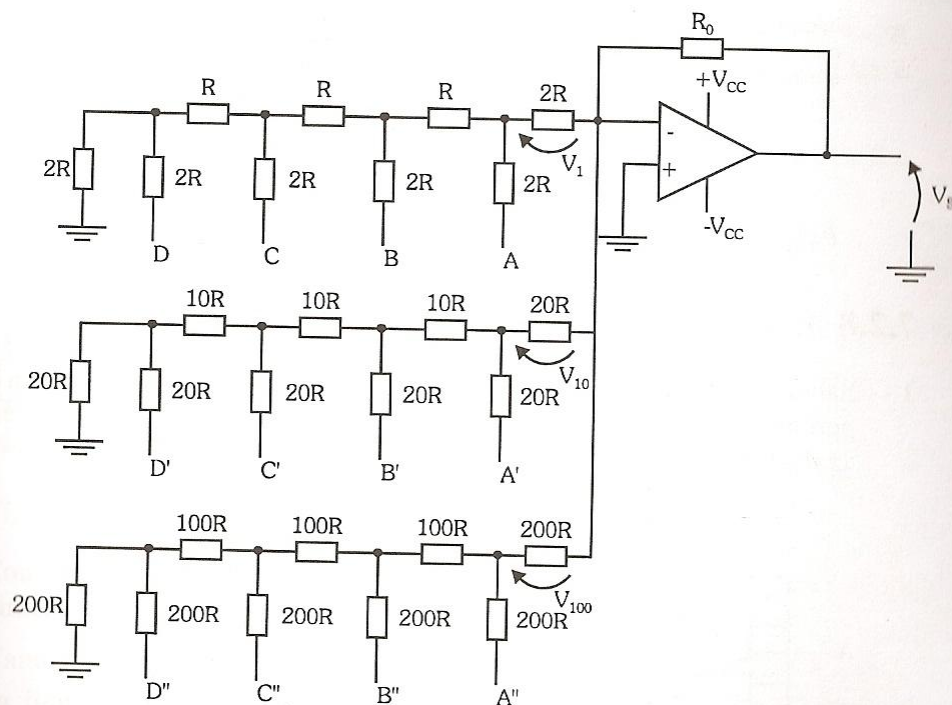



Figura 7.36

Revendo-se o funcionamento do circuito conversor digital-analógico com rede R-2R e do amplificador operacional, pode-se facilmente compreender o seu funcionamento.

A tensão Vs, nesta situação, será dada por:

$$Vs = -\frac{R_0}{2R}.\left(V_1 + \frac{V_{10}}{10} + \frac{V_{100}}{100}\right)$$

7.2.7 Conversão de um Código qualquer para Analógico

Uma maneira simples de convertermos uma informação codificada num código qualquer em uma informação analógica, é a de efetuarmos, primeiramente, a conversão desse código para o código BCD 8421 e, em seguida, efetuarmos a conversão digital-analógica, utilizando um dos processos vistos nos itens precedentes.

A figura 7.37 apresenta a estrutura geral deste processo.

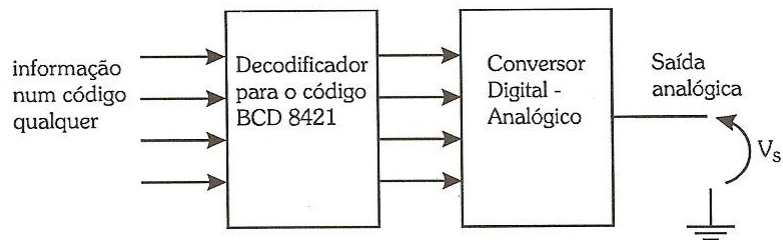



Figura 7.37

7.2.8 Exercícios Resolvidos

1 - Sabendo-se que as portas lógicas do conversor D/A da figura 7.38 pertencem à família TTL (Nível 1 de saída = 5V), calcule as tensões analógicas de saída para as entradas 1010_2 e 1111_2.

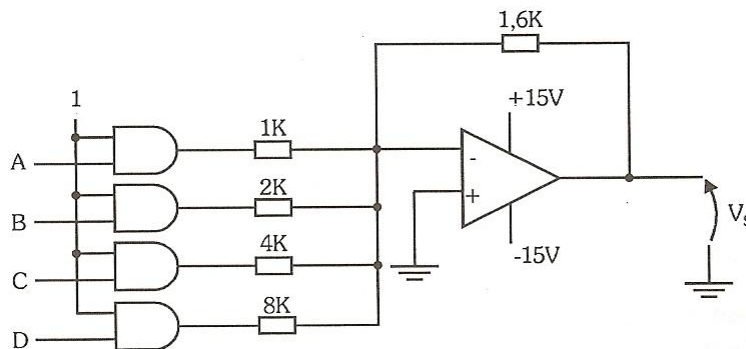



Figura 7.38

Utilizando a expressão geral do conversor D/A com amplificador operacional desenvolvida no item 7.2.2 e os dados do circuito, obtemos o resultado para cada uma das entradas:

entrada 1010_2 =>
$$Vs = -\frac{V.R_0}{R}.\left(A + \frac{B}{2} + \frac{C}{4} + \frac{D}{8}\right)$$

$$Vs = -\frac{5.1600}{1000}.\left(1 + 0 + \frac{1}{4} + 0\right)$$

$$\therefore Vs = -10V$$

entrada 1111_2 =>
$$Vs = -\frac{5.1600}{1000}.\left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8}\right)$$

$$\therefore Vs = -15V$$

2 - Dimensione um conversor D/A com amplificador operacional e rede R-2R para, a partir da entrada binária no código BCD8421, fornecer à saída o nível analógico correspondente. Adote como nível de entrada o valor 5V.

Este conversor, assim especificado, reproduzirá uma saída na faixa de 0 a -9V, sendo seu circuito visto na figura 7.39.

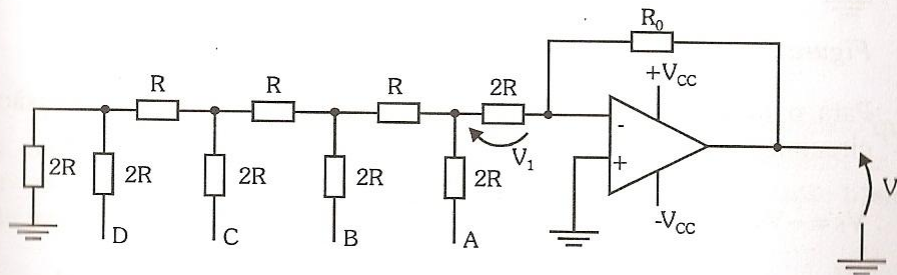



Figura 7.39

Considerando a tensão de saída máxima, no caso igual a -9V, podemos dimensionar a tensão de alimentação em ±9V.

Vamos utilizando a rede R-2R, calcular o valor de V_1 indicado no circuito da figura 7.39 através do caso 1000 aplicado à entrada digital, sendo o nível 1 de entrada a 5V. A figura 7.40 mostra o caso 1000 aplicado à rede R-2R para o cálculo de V_1.

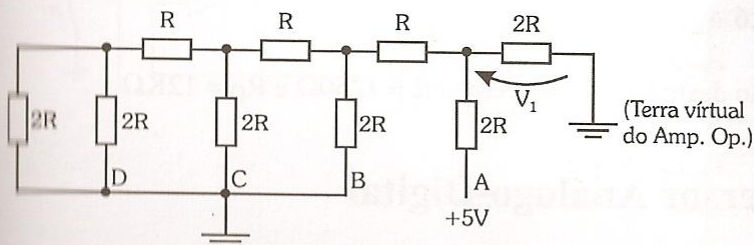
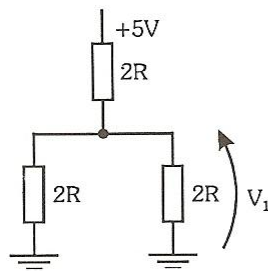



Figura 7.40

Efetuando as associações entre os resistores, obtemos o circuito equivalente visto na figura 7.41 e, através deste, equacionamos e calculamos V_1.

$$V_1 = \frac{5 . R}{3R}$$

$$V_1 = \frac{5}{3}$$

$$V_1 = 1{,}67V$$

Figura 7.41

Para o dimensionamento dos resistores, vamos utilizar a expressão do circuito, concluída no item 7.25:

$$V_S = -V_1 . \frac{R_0}{2R}$$

Uma vez que o caso 1000 equivale à saída analógica igual a -8V, substituindo na expressão, obtemos a relação entre os resistores:

$$-8 = -1{,}67 . \frac{R_0}{2R}$$

$$\frac{16}{1{,}67} = \frac{R_0}{R}$$

$$\therefore \frac{R_0}{R} = 9{,}6$$

Em função deste fator, adotaremos R = 1250Ω e R_0 = 12KΩ.

7.3 Conversor Análogo-Digital

Vimos neste capítulo, a conversão digital-anlógica, mas também existe a necessidade de efetuarmos a conversão reversa, ou seja, a conversão análogo-digital. Vamos estudar a seguir, o circuito que efetua esta conversão.

O processo de conversão análogo-digital consiste, basicamente, em entrarmos com a informação de forma analógica e recolhermos na saída essa mesma informação de forma digital, como esquematizado na figura 7.42.

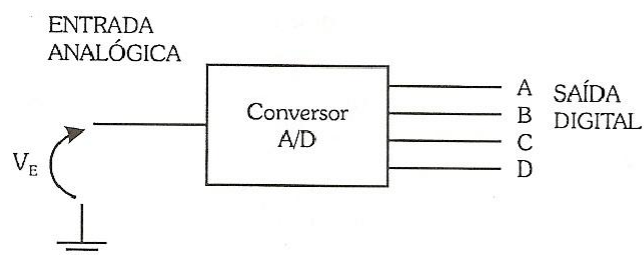



Figura 7.42

O circuito que efetua esta conversão é um pouco mais sofisticado que o dos conversores digital-analógicos, pois necessita de um contador e um conversor digital-analógico para efetuar a conversão. Sua configuração básica é vista na figura 7.43.

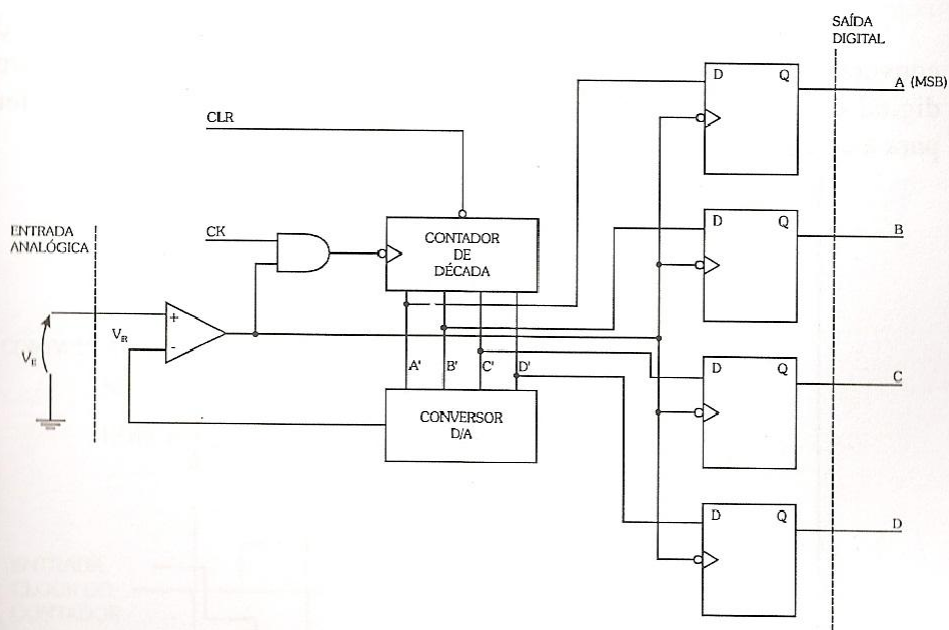



Figura 7.43

O circuito é basicamente constituído por um contador de década que gera o código BCD 8421 nas saídas A', B', C' e D'. Estas saídas são injetadas num conversor digital-analógico, fazendo com que este apresente na saída uma tensão de referência. Esta, por sua vez, é injetada em uma das entradas de um circuito comparador, montado a partir de um amplificador operacional; à outra entrada é injetado o sinal analógico a ser convertido.

A saída deste comparador gera o clock dos flip-flops do circuito de saída e também aciona uma chave digital (porta E), que bloqueará ou não a entrada do clock do contador de década.

Feita esta breve apresentação do circuito, vamos fazer uma análise do funcionamento de cada uma de suas partes integrantes.

O contador de década, que possui um funcionamento por nós já conhecido, apresenta o seguinte diagrama de estados:

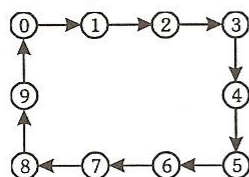



Figura 7.44

A ligação das saídas A', B', C' e D' do contador nas entradas de um conversor digital-analógico, faz com que este transforme esta informação digital em analógica. A tensão de saída do conversor, que serve de referência para a comparação, é mostrada no gráfico da figura 7.45.

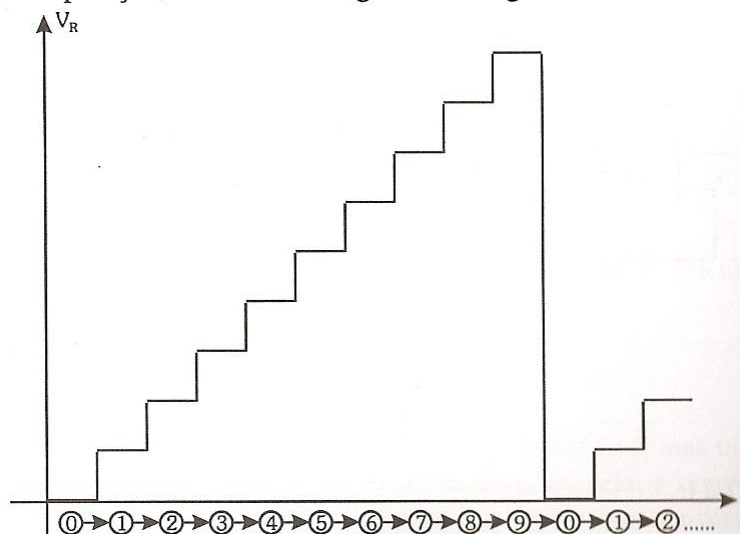



Figura 7.45

O comparador possui na sua entrada não inversora, o sinal analógico a ser convertido (V_E), e na outra entrada, o sinal de referência, fornecido pelo circuito conversor digital-analógico (V_R). A comparação desses sinais resultará na saída do comparador, uma tensão de nível 0, quando V_R for menor que V_E, confome esquematizado na figura 7.46.

V_E — (+), V_R — (-), S

$V_R < V_E \rightarrow S = 1$

$V_R > V_E \rightarrow S = 0$

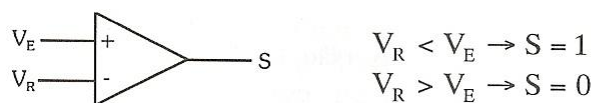



Figura 7.46

A chave digital (porta E) tem em uma entrada o clock, e na outra entrada a saída do comparador. Enquanto a saída do comprador estiver em nível 1 ($V_R < V_E$), a chave dará passagem ao pulso de clock que aciona as mudanças de estado do contador. A partir do momento em que a saída do comparador for para 0, esta chave bloqueará a passagem do clock, fazendo com que o contador permaneça no seu estado que será numericamente igual à tensão de entrada analógica.

Para entendermos o funcionamento do circuito até este ponto, vamos elaborar um exemplo numérico para $V_E = 4V$:

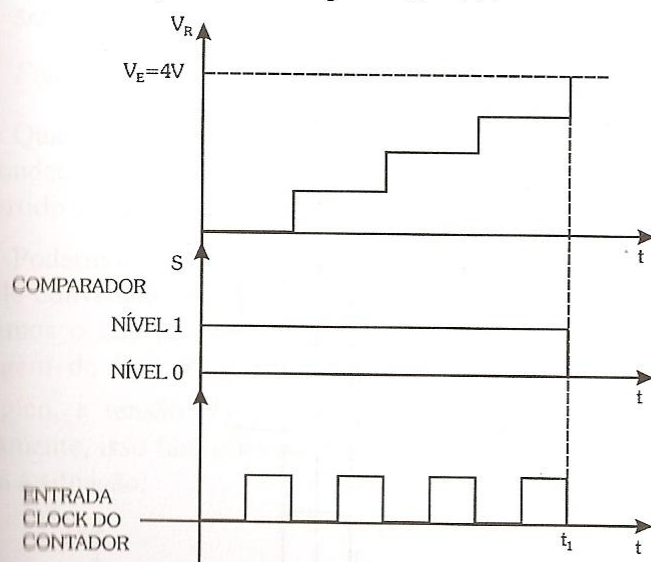



Figura 7.47

Saída do contador a partir do instante t_1:

A'	B'	C'	D'
0	1	0	0

$\Rightarrow 4_{10}$

A saída S do comparador também funciona como clock dos flip-flops tipo D, sendo que, no instante em que S passa de 1 para 0, estes armazenarão a informação contida nas saídas A', B', C' e D', que é o valor codificado da

tensão analógica de entrada. As saídas destes flip-flops permanecerão neste estado até que seja reiniciado o processo.

Para reiniciarmos o processo de conversão, basta aplicarmos um pulso 0 à entrada clear do contador. Isso fará com que este assuma estado 0, fazendo com que V_R retorne a 0, S volte a nível 1 e por fim libere a passagem do clock do contador, reiniciando assim, o processo de conversão do novo valor de V_E.

Os gráficos da figura 7.48 mostram a atuação de cada parte principal do circuito através de um exemplo, onde a tensão analógica de entrada está em 3V e passa para 2V. A função dos flip-flops de saída é a de manter a saída durante a reiniciação do processo, ou seja, quando o contador reinicia a contagem, mantendo, portanto, a situação de saída anterior. Assim que o contador for bloqueado com a nova informação, é dado um pulso 0 de clock nestes flip-flops, obtendo assim a armazenagem desta, permanecendo na saída, até que o comparador forneça um novo pulso de descida.

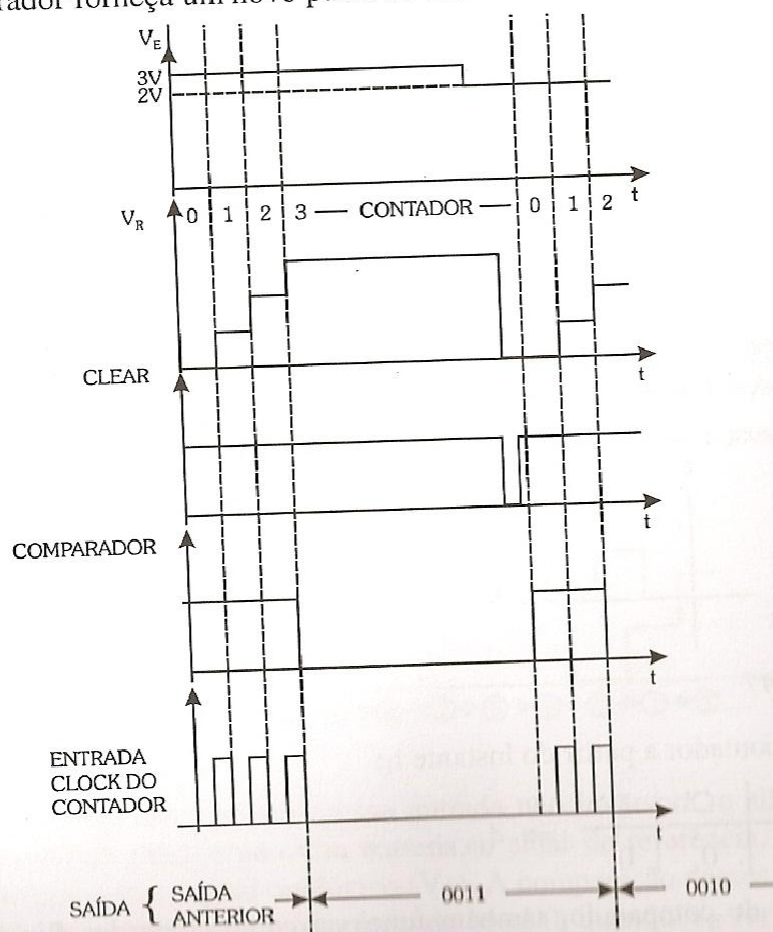



Figura 7.48

Uma das características que deve ser previamente estudada, dependendo da aplicação do circuito, é a sensibilidade, pois o circuito como foi apresentado, arredondará o valor analógico, resultando na saída apenas números inteiros. Como exemplo, tomemos o caso da conversão de uma tensão de 1,2V:

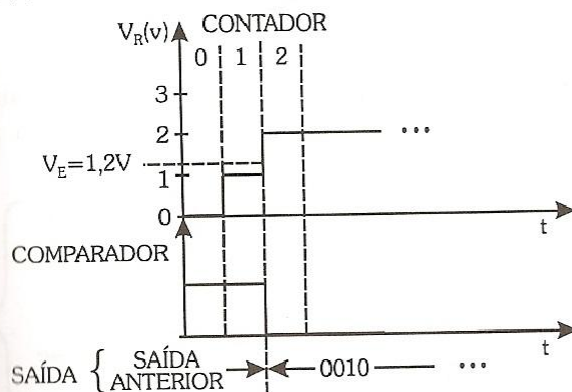



Figura 7.49

Quando a entrada analógica for um valor fracionário, este valor será arredondado para o número imediatamente superior, e na saída, teremos o valor convertido em digital na forma do código BCD 8421.

Podermos perceber que dependendo do valor analógico de entrada, o erro de conversão será elevado. Um meio de solucionarmos este problema, é trocarmos o contador de década por dois contadores, de forma a efetuar a contagem de 0 a 99_{10}. Isso fará com que na saída do conversor digital-analógico, a tensão V_R possua 10 divisões em cada um de seus degraus. Obviamente, isso fará com que o erro seja diminuído. O gráfico da figura 7.50 ilustra a situação:

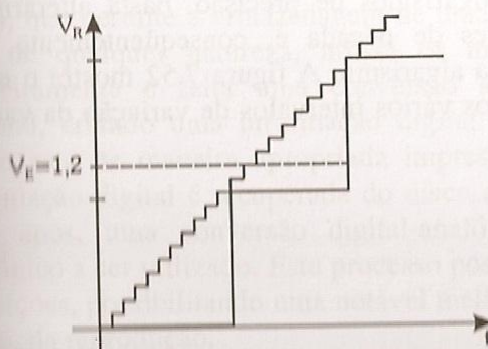



Figura 7.50

No exemplo com V_E = 1,2V, o contador de 0 a 99_{10} irá parar a contagem no estado 12, daí podemos converter a saída do contador do algarismo mais significativo e, separadamente, a do algarismo menos significativo, gerando na saída:

Podemos notar no exemplo, que a conversão apresenta mais um algarismo de precisão. O circuito que efetua esta conversão é visto na figura 7.51 .

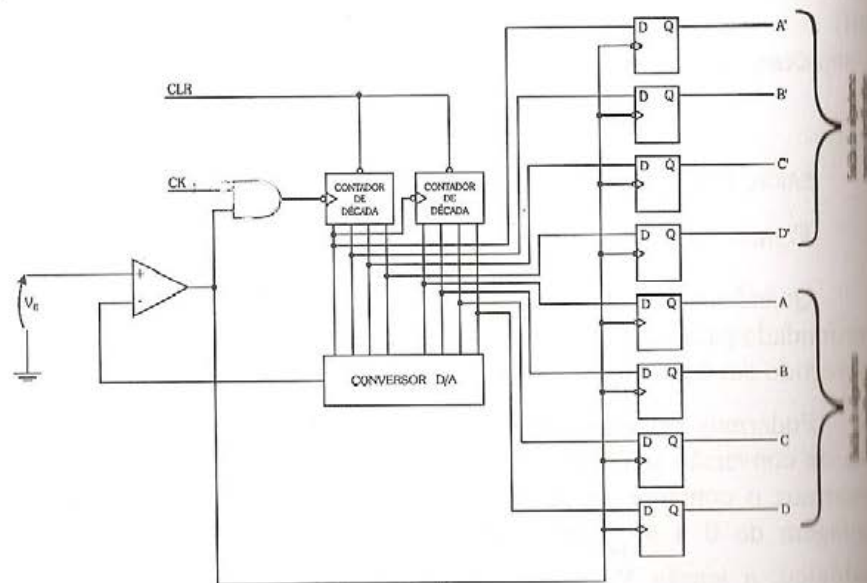



Figura 7.51

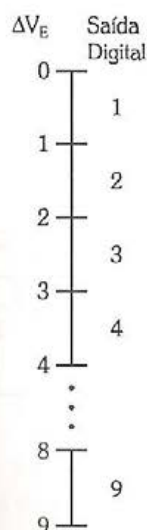
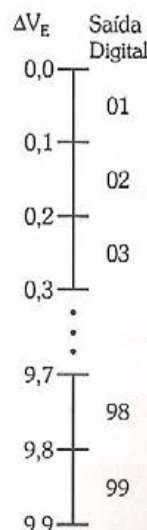
Se necessitarmos de mais algarismos de precisão, basta alterarmos o circuito, inserindo mais contadores de década e, conseqüentemente, mais quatro flip-flops de saída para cada algarismo. A figura 7.52 mostra o estado das saídas do conversor A/D para os vários intervalos de variação da variável de entrada:

(a) Saída Digital com conversão de 1 algarismo em função da faixa de variação V_E.

(b) Saída Digital com conversão de 2 algarismos em função da faixa de variação V_E.

Figura 7.52

As conversões análoga-digitais e digitais-analógicas são largamente utilizadas em processos de medidas, instrumentação e controle eletrônico.

Atualmente, na era digital, mesmo os sinais analógicos estão, predominantemente, sendo processados digitalmente sejam eles de áudio, vídeo ou, ainda, de comunicações, possuindo a vantagem de poderem ser facilmente transmitidos e armazenados.

Em áudio e vídeo, o sistema de gravação e reprodução em **CD (Compact Disc)**, que permite a armazenagem de uma grande quantidade de dados sejam eles de qualquer natureza, utiliza os mesmos processos vistos até aqui: primeiramente é feita uma conversão análoga-digital do sinal analógico original, criando uma informação digital modulada em um trem de pulsos, sendo esta de maneira apropriada impressa no disco. Para a reprodução, a informação digital é recuperada do disco através de leitura a **laser**, sofrendo, logo após, uma conversão digital-analógica, gerando novamente o sinal analógico a ser utilizado. Este processo possui a vantagem de eliminar ruídos e distorções, possibilitando uma notável melhoria na qualidade da gravação, bem como, da reprodução.

7.3.1 Voltímetro Digital

Podemos utilizar o conversor análogo-digital como sendo um voltímetro digital, pois, se na entrada injetamos a tensão a ser medida, nos bits de saída essa tensão será codificada no código BCD 8421. Se na saída digital colocarmos um decodificador do código BCD 8421 para um display de 7 segmentos, poderemos ler o seu valor.

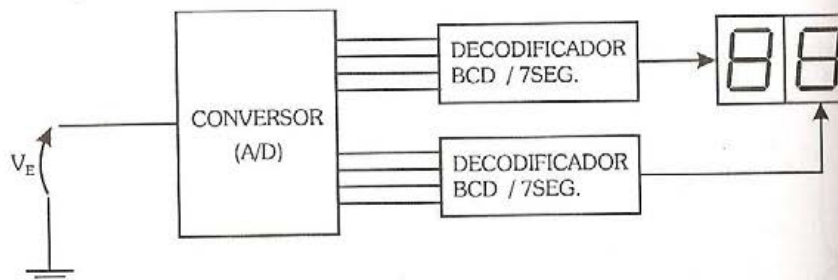



Figura 7.53

7.4 Geradores de Formas de Ondas Digitais

Os geradores de formas de ondas digitais são dispositivos que estão sendo muito difundidos ultimamente. Trata-se da aplicação de alguns dos circuitos vistos até aqui, tais como, contadores e conversores digital-analógicos. O processo de geração de forma de onda é de simples compreensão e nós vamos estudá-lo, esquematizando circuitos para gerar desde formas de onda simples até uma forma de onda qualquer. Uma primeira apresentação em blocos é vista na figura 7.54.

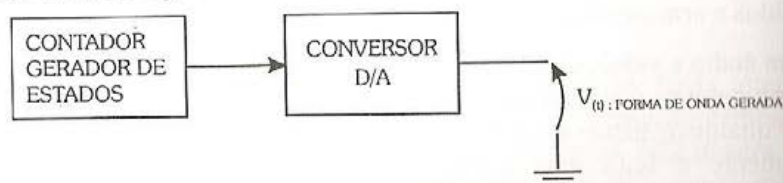



Figura 7.54

7.4.1 Gerador de Rampa Digital

Vamos iniciar com um dos mais simples geradores digitais que é o de Rampa. Utilizamos, neste caso, como contador gerador de estados um de 0 a 8. O circuito é visto na figura 7.55.

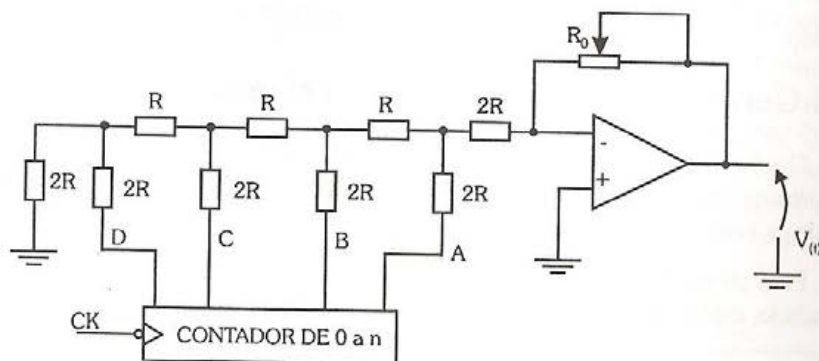



Figura 7.55

Fazendo n igual a 9, teremos um contador de década, sendo a respectiva forma de onda de saída, vista na figura 7.56.

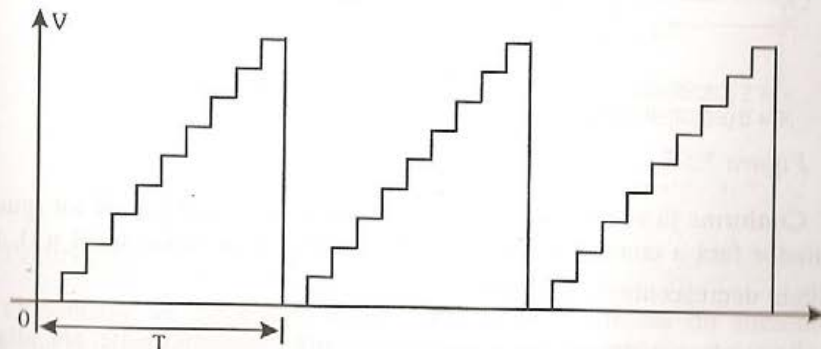



Figura 7.56

Podemos notar que esta forma de onda também é uma aproximação de um sinal do tipo dente de serra. Se quisermos uma definição melhor, basta colocarmos um contador de 0 a n, e sendo n um número maior, isso fará com que tenhamos um maior número de degraus.

Este circuito permite também um controle do valor da amplitude da tensão de saída, bastando para isso alterarmos o ganho do amplificador ($g = R_0 / 2R$). Assim sendo, se aumentarmos R_0, aumentaremos o valor do ganho e, conseqüentemente, o valor da amplitude do sinal e, se diminuirmos R_0, diminuiremos esta amplitude.

Um outro controle que este circuito permite é o de freqüência. Para isso, basta variarmos a freqüência do pulso de clock. Se esta for maior, o período T

será menor e, por conseguinte, a freqüência do sinal será maior. Se a freqüência do pulso de clock for menor, implicará na freqüência de v(t) menor.

7.4.2 Gerador de Forma de Onda Triangular

O processo de obtenção deste é análogo ao anterior, bastando, então, projetarmos um contador que faça inicialmente a contagem crescente e, em seguida, a contagem decrescente.

Para efetuarmos este projeto, vamos utilizar o contador crescente/decrescente, já estudado, cuja esquematização em bloco é vista na figura 7.57.

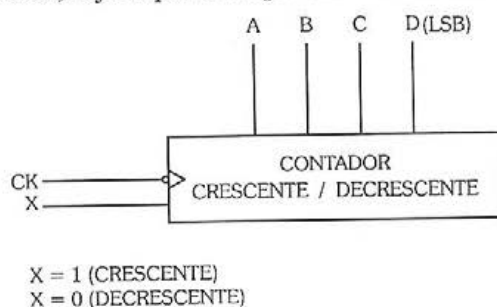



Figura 7.57

Conforme já visto neste circuito, se a entrada de controle X for igual a 1, o contador fará a contagem crescente de 0 a 15_{10} e se X for igual a 0, fará a contagem decrescente de 15_{10} a 0.

Para conseguirmos que o contador conte crescentemente até atingir o estado 15 e, na seqüência, volte decrescentemente até o estado 0, é necessário acrescentar o circuito de controle visto na figura 7.58.

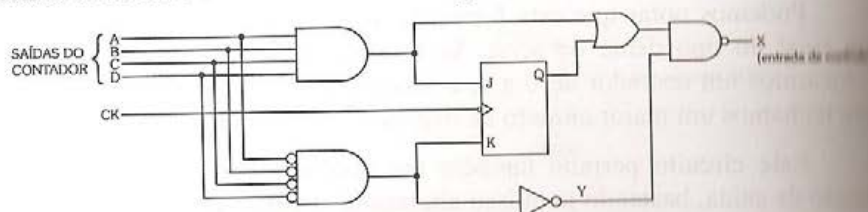



Figura 7.58

Quando o contador estiver no estado 0, o ponto Y, que em todos os outros casos é igual a 1, estará em 0, e as entradas J e K do flip-flop de controle serão 0 e 1 respectivamente, impondo o estado seguinte igual a 0 na saída Q. Estando X em 1, o contador fará a contagem crescente, e durante a passagem

de todos os outros estados, as entradas J e K permanecerão em 0, o que manterá a entrada X no contador em 1, continuando a contagem crescente. O contador, ao atingir o estado 15, fará com que as entradas J e K do flip-flop de controle sejam 1 e 0 respectivamente, forçando, assim, o estado seguinte da saída Q para 1. Em conseqüência disso, X será, até chegar o estado 0, igual a 0 e o contador irá continuar a contagem decrescente. Ao chegar o estado 0, recomeçará como já explicado, a contagem crescente. Assim, teremos este contador executando a contagem crescente, após a decrescente e assim sucessivamente.

O circuito gerador de tensão triangular é visto na figura 7.59.

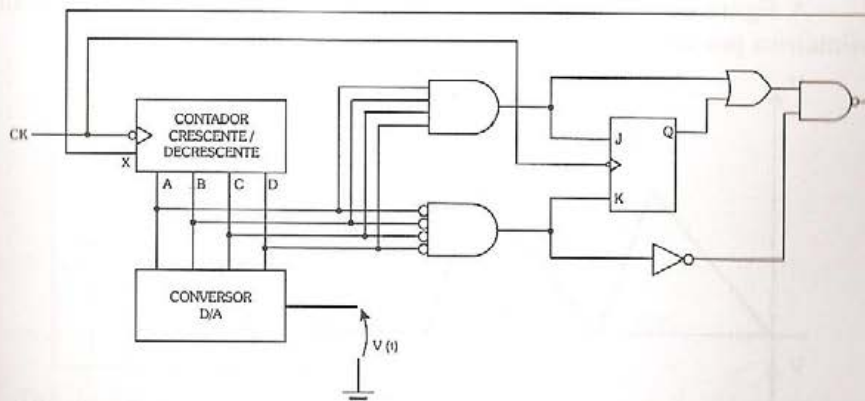



Figura 7.59

O controle de amplitude dessa tensão é feito através do aumento ou diminuição do ganho do amplificador, e o controle de freqüência é feito através da variação da freqüência de clock, análogo ao circuito gerador de rampa, anteriormente visto.

A figura 7.60 mostra a forma de onda analógica obtida na saída do circuito.

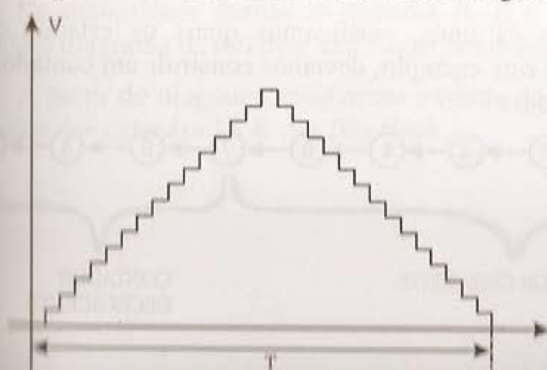



Figura 7.60

Podemos notar que, para qualquer forma de onda, se aumentarmos o número de degraus, mais próximos da forma de onda estaremos, porém maior será a freqüência de clock de que iremos necessitar.

7.4.3 Gerador de uma Forma de Onda qualquer

Podemos gerar uma forma de onda qualquer com geradores de formas de onda digitais. Para isso devemos, primeiramente, digitalizar a forma de onda a qual queremos gerar e, em seguida, executar o projeto conforme o processo mostrado no exemplo a seguir.

A figura 7.61 mostra a digitalização de uma forma de onda triangular assimétrica por oito estados (contador de 3 bits).

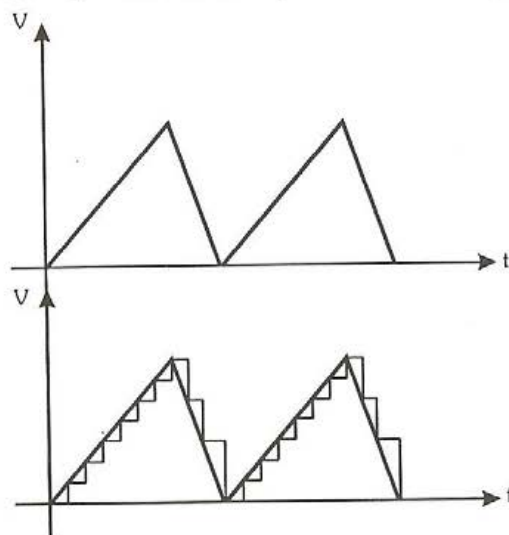



Figura 7.61

Após a digitalização da onda, verificamos quais os estados que o contador deve assumir. Para este exemplo, devemos construir um contador com o seguinte diagrama de estados:

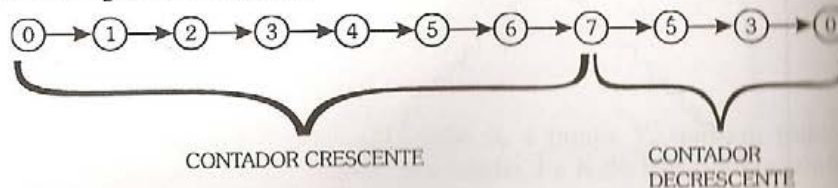



Figura 7.62

Para que o diagrama de estados seja executado pelo contador corretamente, utilizamos uma variável auxiliar X. Assim sendo, montamos a tabela da verdade:

X	A	B	C	
0	0	0	0	
0	0	0	1	X = 0 → Crescente
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	1	0	1	X = 1 → Decrescente
1	0	1	1	

Tabela 7.4

Considerando essa variável auxiliar, o diagrama de estados passa a ser:

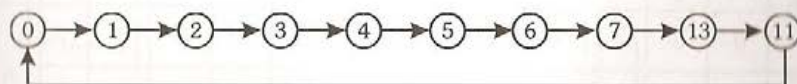



Figura 7.63

Se analisarmos apenas as entradas A, B e C, veremos que o contador executa o diagrama de estados, visto anteriormente.

A partir do diagrama, montamos a tabela da verdade do contador com as situações das entradas J e K dos flip-flops.

X	A	B	C	J_x	K_x	J_A	K_A	J_B	K_B	J_C	K_C
0	0	0	0	0	X	0	X	0	X	1	X
0	0	0	1	0	X	0	X	1	X	X	1
0	0	1	0	0	X	0	X	X	0	1	X
0	0	1	1	0	X	1	X	X	1	X	1
0	1	0	0	0	X	X	0	0	X	1	X
0	1	0	1	0	X	X	0	1	X	X	1
0	1	1	0	0	X	X	0	X	0	1	X
0	1	1	1	1	X	X	0	X	1	X	0
1	1	0	1	X	0	X	1	1	X	X	0
1	0	1	1	X	1	0	X	X	1	X	1

Tabela 7.5

Da tabela da verdade tiramos as expressões simplificadas de J e K:

J_X:

		$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}$	$\overline{A}$	0	0	0	0
$\overline{X}$	A	0	0	1	0
X	A	X	X	X	X
X	$\overline{A}$	X	X	X	X

(a) $J_x = ABC$

K_X:

(b) $K_x = B$

J_A:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}\,\overline{A}$	0	0	1	0
$\overline{X}A$	X	X	X	X
XA	X	X	X	X
$X\overline{A}$	X	X	0	X

(c) $J_A = BC\overline{X}$

K_A:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}\,\overline{A}$	X	X	X	X
$\overline{X}A$	0	0	0	0
XA	X	1	X	X
$X\overline{A}$	X	X	X	X

(d) $K_A = X$

J_B:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}\,\overline{A}$	0	1	X	X
$\overline{X}A$	0	1	X	X
XA	X	1	X	X
$X\overline{A}$	X	X	X	X

(e) $J_B = C$

K_B:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}\,\overline{A}$	X	X	1	0
$\overline{X}A$	X	X	1	0
XA	X	X	X	X
$X\overline{A}$	X	X	1	X

(f) $K_B = C$

J_C:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}\,\overline{A}$	1	X	X	1
$\overline{X}A$	1	X	X	1
XA	X	X	X	X
$X\overline{A}$	X	X	X	X

(g) $J_C = 1$

K_C:

	$\overline{B}\,\overline{C}$	$\overline{B}C$	BC	$B\overline{C}$
$\overline{X}\,\overline{A}$	X	1	1	X
$\overline{X}A$	X	1	0	X
XA	X	0	X	X
$X\overline{A}$	X	X	1	X

(h) $K_C = \overline{A} + \overline{B}.\overline{X}$

Figura 7.64

O esquema do contador é visto na figura 7.65.

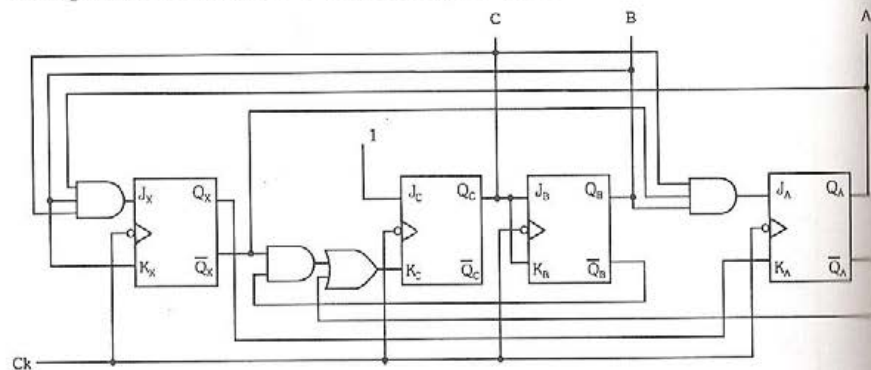



Figura 7.65

Na figura 7.66, temos o diagrama de blocos deste gerador da forma de onda triangular assimétrica.

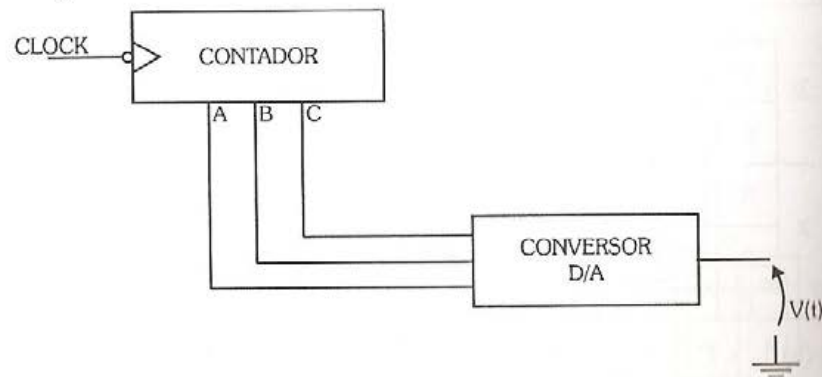



Figura 7.66

Seguindo o processo visto no exemplo anterior, podemos esquematizar um gerador de uma forma de onda qualquer, bastando para isso projetar o contador conveniente e ligar suas saídas às entradas de um conversor digital-analógico.

7.4.4 Exercícios Resolvidos

1 - Esboce um ciclo completo da forma de onda às saídas do conversor D/A visto na figura 7.67, sabendo-se que o contador é da família TTL (nível de saída = 5V) e que a freqüência de clock é 1 MHz.

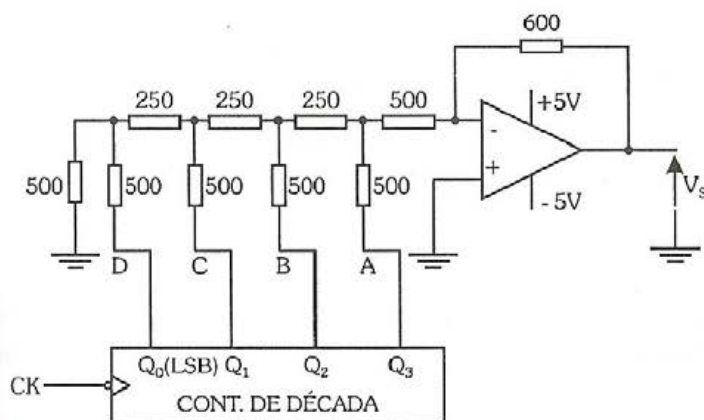



Figura 7.67

O primeiro passo para a solução é o cálculo do período do sinal de clock, para determinarmos o intervalo de tempo de cada estado de saída do contador, e assim obtermos a graduação do eixo de tempo do sinal de saída. Assim sendo, temos:

$$T = \frac{1}{f} \rightarrow T = \frac{1}{1.10^6} = 1\mu s$$

$$\therefore \Delta t = 1\mu s$$

Cabe aqui ressaltar que a duração de cada estado de saída do contador é igual a um período, pois cada saída assume o novo estado na descida do pulso de clock, após decorrido um ciclo completo deste.

O passo seguinte é o de calcular a tensão de entrada no amplificador operacional, isto é, a queda de tensão no último resistor de 500Ω (V_1). Para tanto, vamos utilizar a entrada 1000 que equivale ao estado 8. O circuito simplificado com esta entrada aplicada é visto na figura 7.68.

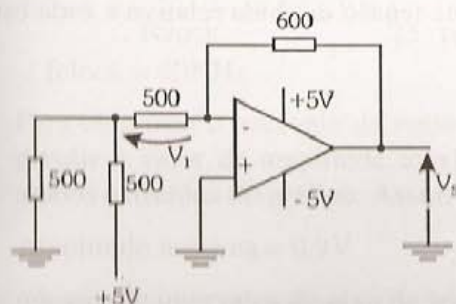



Figura 7.68

A tensão V_1 será:

$$V_1 = \frac{Vcc}{3} \rightarrow V_1 = \frac{5}{3} = 1{,}67V$$

Aplicando a expressão de saída do amplificador operacional, já visto, obtemos:

$$Vs = -V_1 . \frac{R_0}{2R} \rightarrow Vs = \frac{-1{,}67 . 600}{500} = -2V$$

Dividindo esta por 8, obtemos o intervalo de tensão de saída relativo a cada estado, obtendo assim a graduação do outro eixo. Assim sendo, temos:

$$\Delta V = \frac{-2}{8} = -0{,}25V$$

De posse dos intervalos de cada eixo, construímos o gráfico da saída analógica, visto na figura 7.69.

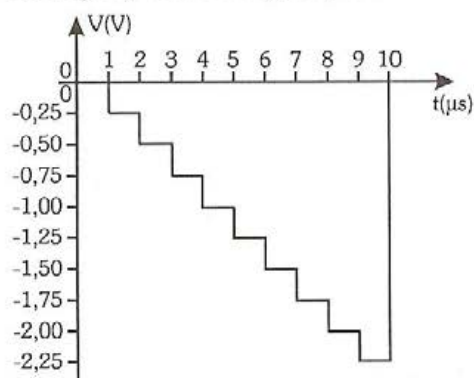



Figura 7.69

2 - Determine a freqüência de clock necessária para gerar o sinal digitalizado visto na figura 7.70, e o intervalo de tensão de saída relativo a cada estado do contador.

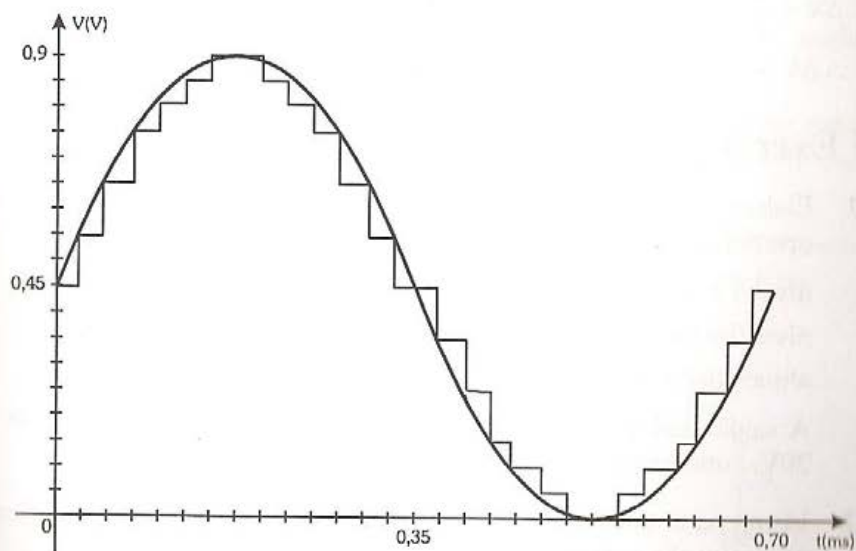



Figura 7.70

Para determinarmos a freqüência de clock, vamos extrair do gráfico o período do sinal e dividi-lo pelo número de intervalos de tempo ocupados pelo sinal digitalizado constantes no gráfico para, assim, obter o período relativo à forma de onda de clock. Assim sendo, temos:

Tsinal = 0,70ms

número de intervalos de tempo = 28

$$Tclock = \frac{0{,}7 \cdot 10^{-3}}{28} = 25 \cdot 10^{-6} = 25\mu s$$

Determinando a freqüência de clock, temos:

$$fclock = \frac{1}{Tclock} \rightarrow fclock = \frac{1}{25 \cdot 10^{-6}} = 40KHz$$

$\therefore$ fclock = 40KHz

Para obtermos o intervalo de tensão relativo a cada estado de saída, basta dividir o valor de amplitude máxima, pelo número de intervalos, sendo ambos extraídos do gráfico. Assim sendo, temos:

Amplitude máxima = 0,9V

número de intervalos do eixo de tensão = 18

$$\Delta V = \frac{0,9}{18} = 0,05V$$

$$\therefore \Delta V = 0,05V$$

7.5 Exercícios Propostos

7.5.1 Elabore um conversor digital-analógico, utilizando amplificador operacional, com as características:

nível 1 = 5V

nível 0 = 0V

alimentação: +15V/-15V

A saída analógica deverá ser lida na escala de um voltímetro de 0 a 20V, com entrada digital variando de 0 a 15_{10}.

7.5.2 Idem, para um conversor digital-analógico, utilizando amplificador operacional e rede R-2R.

7.5.3 Elabore um conversor digital-analógico, utilizando amplificador operacional, com as características:

nível 1 = 5V

nível 0 = 0V

alimentação do operacional: +10V / -10V

A saída analógica deverá ser lida na escala de 0 a 10V de um voltímetro, com entrada digital variando de 0 a 99_{10} em 2 algarismos do código BCD 8421.

7.5.4 Idem, utilizando um conversor digital-analógico com amplificador operacional e rede R-2R.

7.5.5 Desenhe o esquema de um conversor D/A com amplificador operacional com entrada digital para 8 bits. Escreva a expressão geral deste circuito.

7.5.6 No circuito do exercício anterior, adotando R = 1KΩ, R_0 = 1,5 KΩ, nível1 = 5V e Vcc = ± 15V, calcule a tensão de saída para as seguintes entradas digitais:

a) $1F_{16}$ **c)** AB_{16}

b) 56_{16} **d)** FF_{16}

7.5.7 Desenhe a estrutura de um voltímetro digital de 0 a 9,9V, de modo que a tensão de saída seja escrita em display de 2 dígitos.

7.5.8 Utilizando a estrutura obtida no exercício anterior, coloque todos os níveis de entrada e saída dos blocos para a medição de uma entrada analógica igual a 3,2V.

7.5.9 Esboce a forma de onda na saída para o circuito da figura 7.71, sendo o nível 1 do contador igual a 5V e a freqüência de clock 200 KHz.

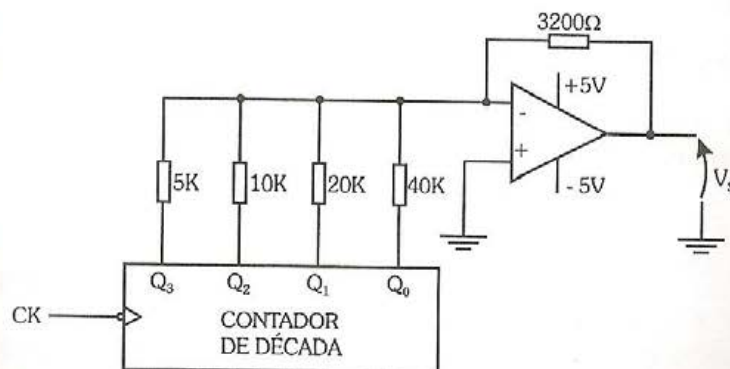



Figura 7.71

7.5.10 Esquematize o circuito completo para gerar a forma de onda mostrada na figura 7.72. Determine a freqüência de clock necessária.

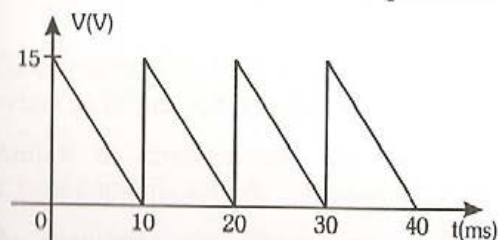



Figura 7.72

7.5.11 Projete o circuito para gerar a forma de onda vista na figura 7.73. Determine a freqüência de clock necessária.

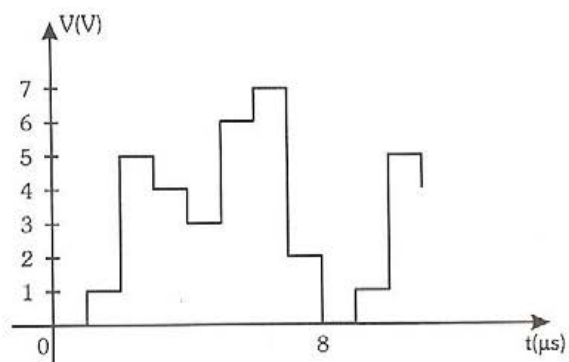
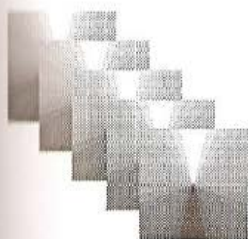



Figura 7.73

7.5.12 Para o sinal do exercício resolvido (2) do item 7.4.4 (figura 7.70), determine o diagrama de estados do contador.

CAPÍTULO 8

Circuitos Multiplex, Demultiplex e Memórias

8.1 Introdução

Neste capítulo, vamos falar de assuntos de grande importância. Trata-se do **Multiplex, do Demultiplex e das Memórias,** utilizáveis em circuitos com microprocessadores.

Os circuitos multiplex são utilizados nos casos em que necessitamos enviar em certo número de informações, contidas em vários canais, a um só canal.

Os circuitos demultiplex efetuam a função inversa à dos multiplex, ou seja, enviam as informações, vindas de um único canal, a vários canais.

Ambos os circuitos são largamente empregados dentro de sistemas digitais, bem como na área de Transmissão de dados.

As memórias são blocos que armazenam informações codificadas digitalmente. Dividem-se basicamente em dois grupos: as memórias de escrita e leitura e as memórias apenas de leitura. Têm sua grande aplicação em sistemas digitais, utilizandos principalmente na área de Informática.

Vamos iniciar o capítulo, desenvolvendo alguns conceitos básicos que serão utilizados nos tipos de circuitos mencionados, no que se refere ao endereçamento.

8.2 Geração de Produtos Canônicos

Como foi visto no capítulo 2, com **n** variáveis booleanas podemos fazer 2^n combinações. Por exemplo, com 2 variáveis podemos formar $2^2 = 4$ possibilidades, sendo estas:

0) $\overline{A} . \overline{B} \rightarrow A = 0$ e $B = 0$

1) $\overline{A} . B \rightarrow A = 0$ e $B = 1$

2) $A . \overline{B} \rightarrow A = 1$ e $B = 0$

3) $A . B \rightarrow A = 1$ e $B = 1$

Vamos considerar a expressão referente ao caso 0: $P_0 = \overline{A} . \overline{B}$. Este produto será igual a 1 somente quando A = B = 0.

No caso 1, temos: $P_1 = \overline{A} . B$, que será igual a 1 somente quando A = 0 e B = 1.

No caso 2, temos: $P_2 = A . \overline{B}$, que será igual a 1 somente quando A = 1 e B = 0.

No caso 3, temos: $P_3 = A . B$, que será igual a 1 somente quando A = 1 e B = 1.

Estes quatro produtos possíveis com 2 variáveis são denominados **produtos canônicos.** Então, com **n** variáveis, temos 2^n produtos canônicos.

8.2.1 Circuito Básico Gerador de Produtos Canônicos

Podemos esquematizar circuitos para gerar produtos canônicos. Um primeiro e mais simples de ser entendido é o constituído por portas E e inversores. A figura 8.1 mostra um exemplo para 2 variáveis de entrada.

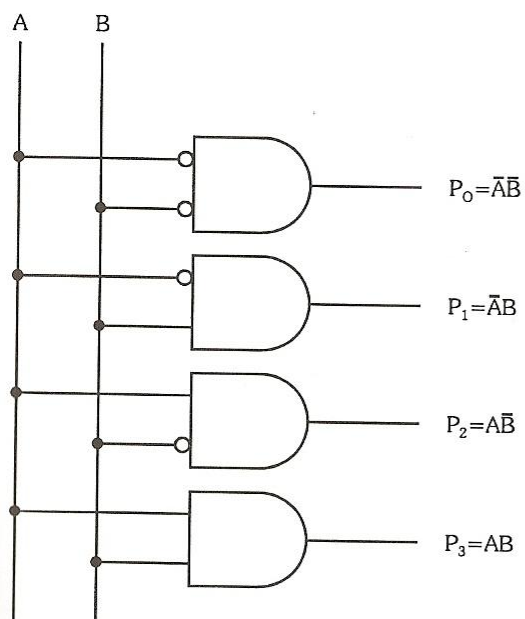



Figura 8.1

Seguindo o mesmo esquema básico, para 3 variáveis, temos o circuito da figura 8.2 visto na página seguinte.

Analogamente, se quisermos gerar os produtos canônicos com **n** variáveis, necessitamos, então, de **2ⁿ** portas E de **n** entradas cada.

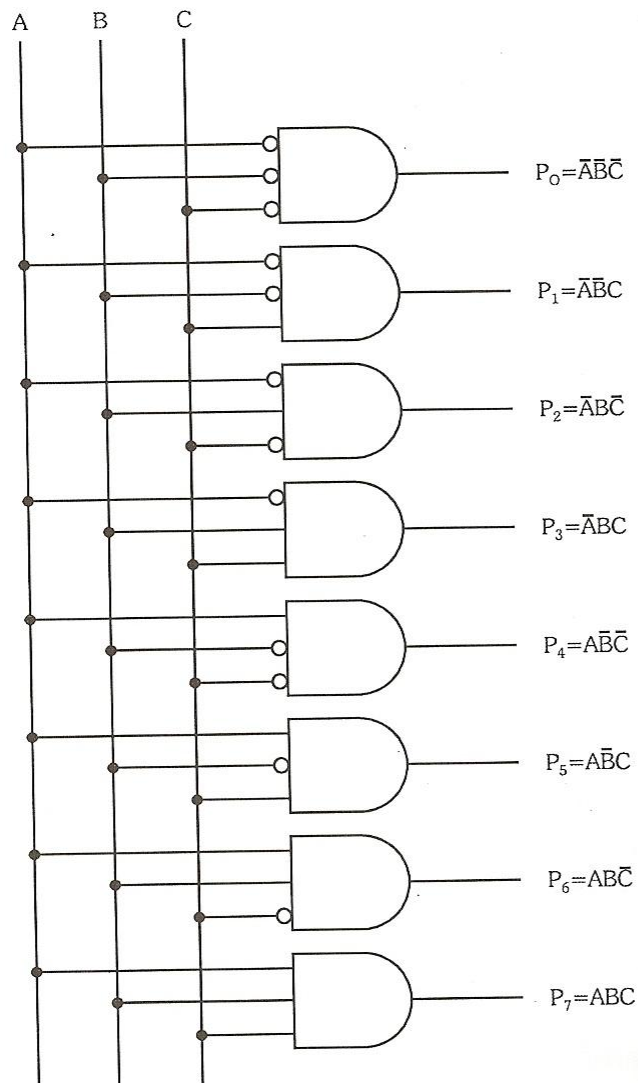



Figura 8.2

8.2.2 Matriz de Simples Encadeamento

Um segundo processo de geração de produtos canônicos é o conhecido como **Matriz de Simples Encadeamento**, que utiliza somente portas E de 2 entradas. O circuito no caso de 2 variáveis, é idêntico ao já visto, utilizando 4 portas E de 2 entradas. Para 3 variáveis, temos o circuito mostrado na figura 8.3.

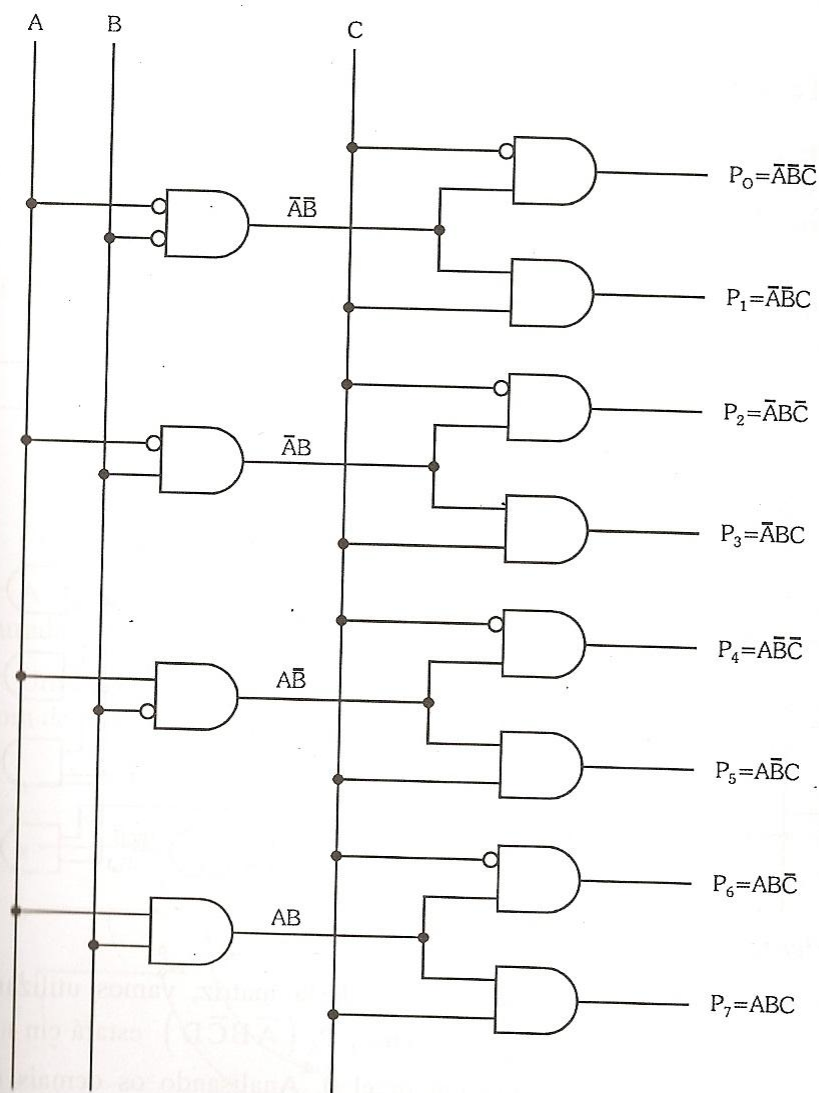



Figura 8.3

Notamos que este circuito foi desenvolvido a partir do circuito de 2 variáveis, visto no item anterior. Se quisermos montar um gerador de produtos canônicos de 4 variáveis, basta colocarmos 2 portas E com entradas $\overline{D}$ e D, respectivamente, em cada saída do circuito de 3 variáveis e assim, sucessivamente, para maior número de variáveis.

Para n variáveis, temos N portas de 2 entradas onde $N = 2^{n+1} - 4$.

Este tipo de matriz é também conhecido como **piramidal.**

8.2.3 Matriz de Duplo Encadeamento

O terceiro processo, que é o mais utilizado por apresentar uma rápida resposta com um menor número de portas E, é conhecido como **Matriz de Duplo Encadeamento.** Este tipo de matriz é muito importante pelo fato de ser utilizado em circuitos multiplex e na estrutura de algumas memórias.

Vamos construir uma matriz de duplo encadeamento para a geração de produtos canônicos de 4 variáveis.

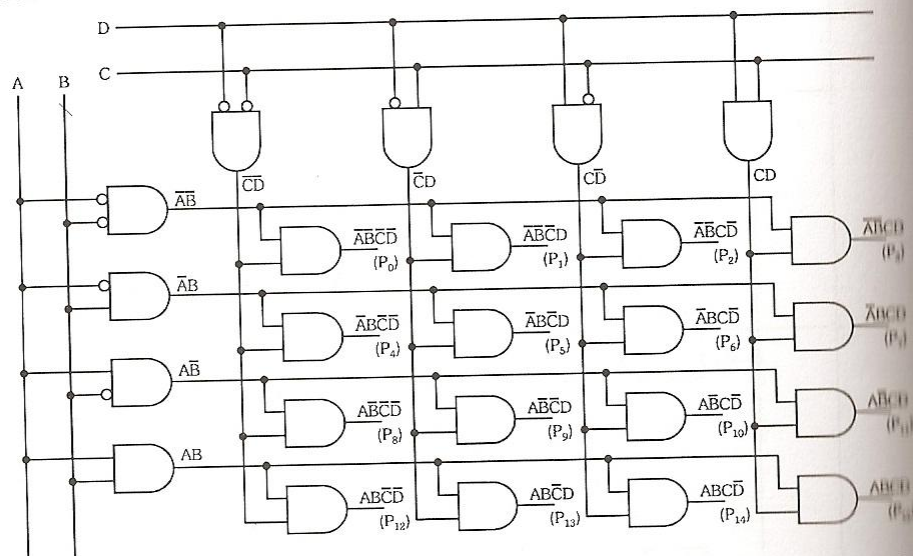



Figura 8.4

Para entendermos o funcionamento desta matriz, vamos utilizar, por exemplo, a entrada 5_{10} $(0101)_2$. Neste caso, P_5 $\left(\overline{A}B\overline{C}D\right)$ estará em nível 1 e todas as demais saídas estarão em nível 0. Analisando os demais casos, veremos que cada um apresentará uma saída 1 para uma entrada específica.

8.3 Multiplex

Como dissemos no início deste capítulo, o circuito multiplex é utilizado para enviarmos as informações contidas em vários canais (fios), a um só canal (fio). Esquematizando o bloco multiplex, temos:

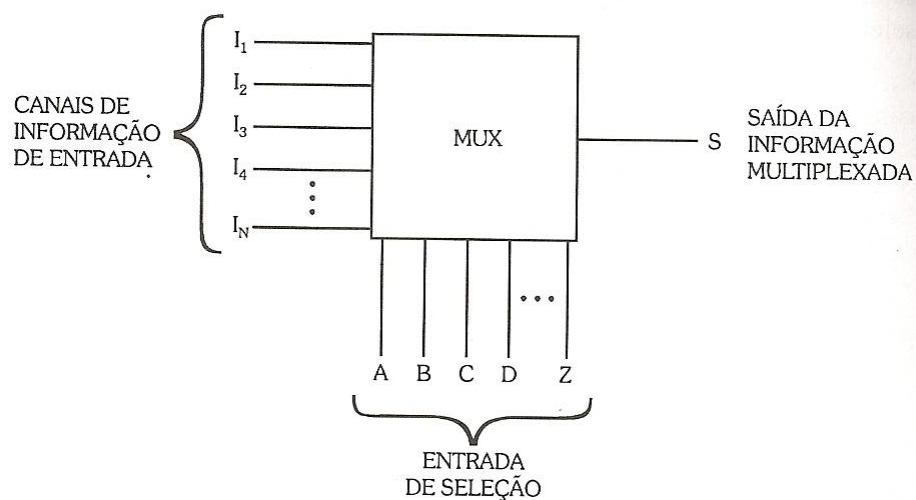



Figura 8.5

A entrada de seleção tem como finalidade escolher qual das informações de entrada, ou qual dos canais de informações deve ser ligado à saída.

Um circuito elementar que efetua uma multiplexação é uma chave seletora de 1 pólo e n posições, esquematizada na figura 8.6.

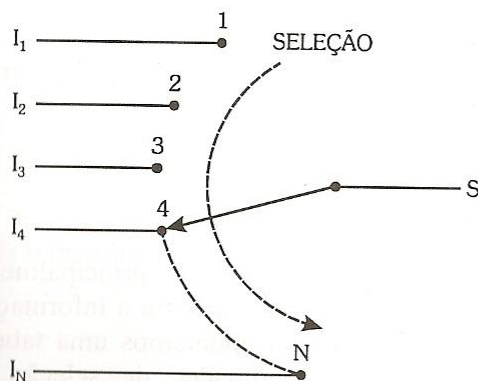



Figura 8.6

Se quisermos ligar, por exemplo, a informação I_1 na saída, basta selecionarmos a posição 1 da chave seletora. Se quisermos conectar à saída a informação I_2, selecionamos a posição 2 e assim, sucessivamente.

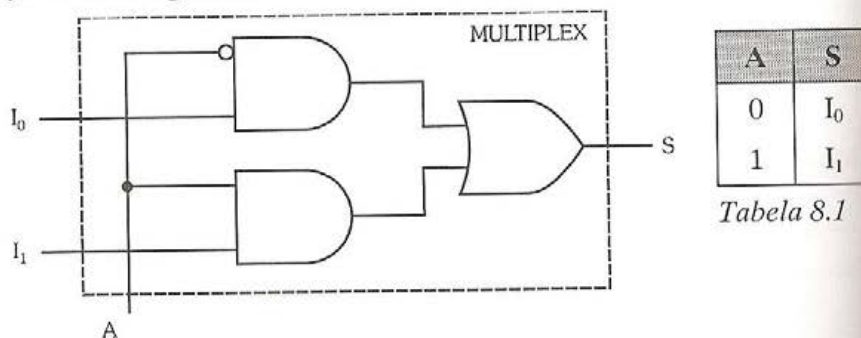
Este processo é o funcionamento básico de um multiplex, sendo que as entradas de seleção irão indicar qual a informação a ser conectada à saída, ou

seja, no exemplo, as variáveis de seleção irão comutar a posição da chave seletora.

O circuito lógico básico que efetua a função de um multiplex de 2 canais, é visto na figura 8.7.

Figura 8.7

A	S
0	I_0
1	I_1

Tabela 8.1

No caso do multiplex básico para 2 informações de entradas I_0 e I_1, temos uma variável de seleção (A). Quando A for igual a 0, teremos na saída, a mesma informação que a entrada I_0 ; se I_0 for igual a 0, S será igual a 0 e se I_0 for igual a 1, S será igual a 1. Neste caso, a informação I_1 será bloqueada pela porta E referente a I_1,pois o outro terminal desta estará ligado em A que valerá 0.

Quando A for igual a 1, I_0 será bloqueado e, analogamente, a informação I_1 aparecerá na saída.

8.3.1 Projeto do Circuito de um Multiplex

Para projetarmos um multiplex, devemos relacionar, principalmente, a possibilidade de que as entradas de seleção irão assumir com a informação de entrada que deve ser conectada à saída. Para isso, montamos uma tabela da verdade onde serão colocadas todas as possibilidades de seleção e as respectivas informações que devem aparecer na saída.

Para mostrarmos passo a passo a elaboração de multiplex, vamos iniciar, efetuando o projeto de um multiplex de 4 canais ou entradas de informações.

Para que possamos conectar aleatoriamente 4 entradas à saída, necessitamos de 2 variáveis de seleção. Com isso, podemos montar a tabela da verdade:

Variáveis de Seleção		Saída
A	B	S
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Tabela 8.2

Montando a tabela, relacionamos os valores assumidos pela saída para cada possibilidade das variáveis de seleção, obtendo, a partir disso, o respectivo produto canônico.

Variáveis de Seleção:	**Situação na saída:**
Caso 0 0 $(P_0 = \overline{A} . \overline{B})$	$S = I_0$
Caso 0 1 $(P_1 = \overline{A} . B)$	$S = I_1$
Caso 1 0 $(P_2 = A . \overline{B})$	$S = I_2$
Caso 1 1 $(P_3 = A . B)$	$S = I_3$

Em função destas expressões, esquematizamos o circuito. A figura 8.8 mostra o circuito obtido do multiplex de 4 canais proposto.

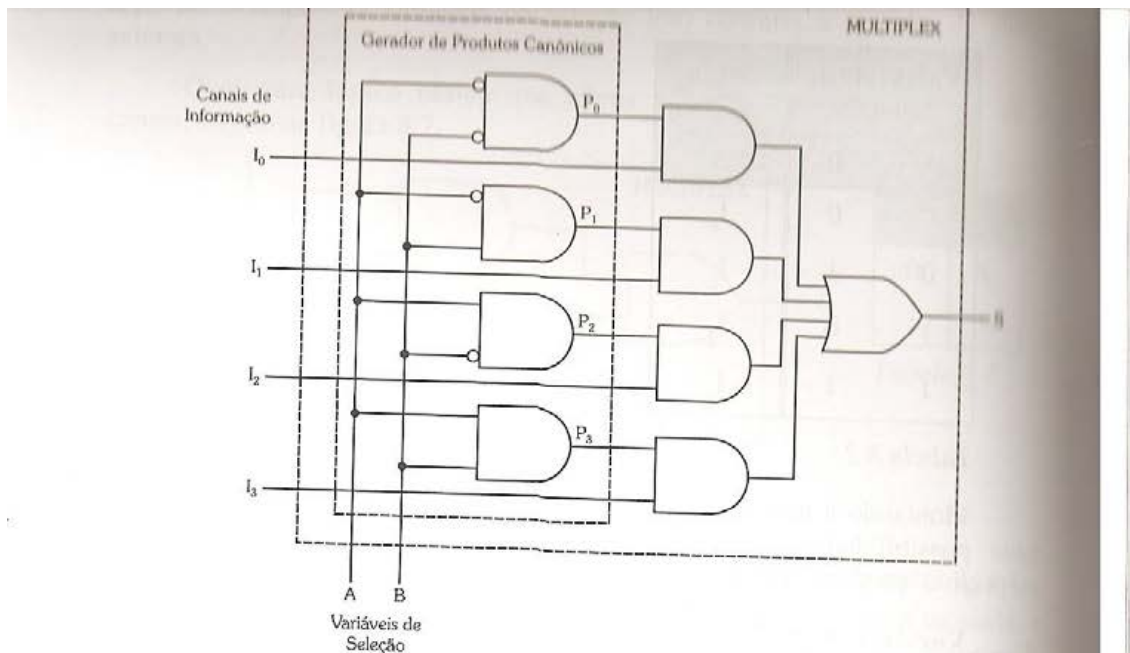



Figura 8.8

Para entendermos o funcionamento do circuito, vamos analisar um dos casos possíveis, por exemplo, o caso em que as variáveis de seleção estiverem na condição A = 1 e B = 0.

Quando ocorrer este caso, o gerador de produtos canônicos apresentará $P_2 = 1$, com isso, a porta E ligada à saída P_2, estará com um dos terminais em nível lógico 1, logo, na sua saída, teremos o valor que I_2 assumir, ou seja, se I_2 for igual a 0, a saída desta será 0; se I_2 for igual a 1, a saída será 1. Sabendo-se que neste caso todas as outras entradas da porta OU estarão em 0, concluímos que, quando as variáveis de seleção estiverem na condição 10 $\left(A\overline{B}\right)$, S será igual a I_2. Para analisarmos os outros casos, basta procedermos de forma análoga.

O circuito foi esquematizado dessa maneira para maior compreensão, normalmente, é representado como mostra a figura 8.9.

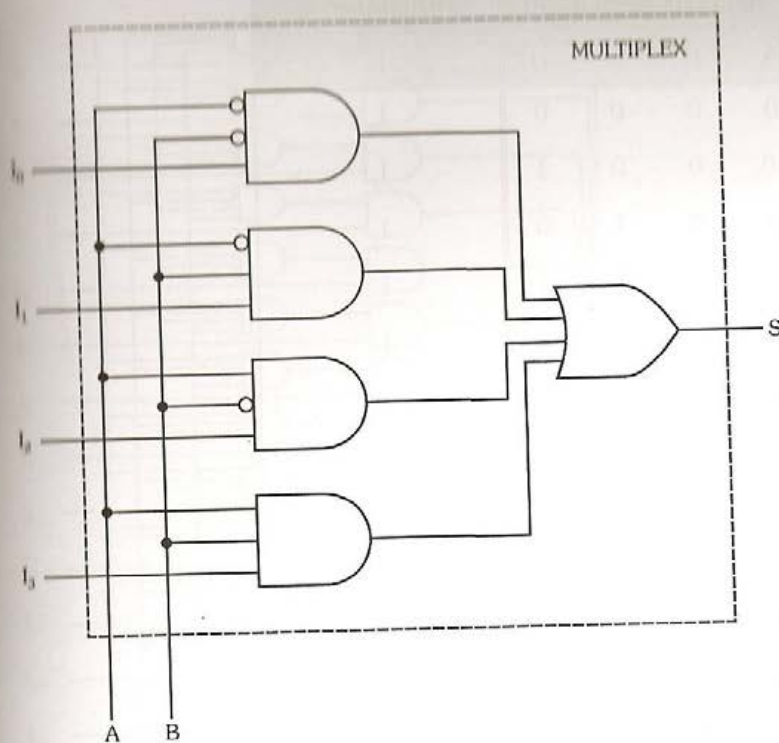



Figura 8.9

Representando o multiplex obtido em bloco, temos:

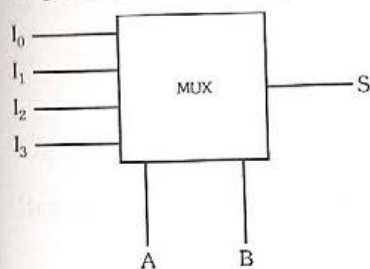



Figura 8.10

Vamos agora, como exemplo, elaborar o circuito de um multiplex de 16 canais segundo o mesmo processo. Para comutarmos 16 entradas necessitamos de 4 variáveis de seleção. O número de informações que as entradas de seleção podem comutar é 2^n, onde **n** é o número de entradas de seleção. Assim sendo, montamos a tabela da verdade:

Variáveis de Seleção				Informações
A	B	C	D	S
0	0	0	0	I_0
0	0	0	1	I_1
0	0	1	0	I_2
0	0	1	1	I_3
0	1	0	0	I_4
0	1	0	1	I_5
0	1	1	0	I_6
0	1	1	1	I_7
1	0	0	0	I_8
1	0	0	1	I_9
1	0	1	0	I_{10}
1	0	1	1	I_{11}
1	1	0	0	I_{12}
1	1	0	1	I_{13}
1	1	1	0	I_{14}
1	1	1	1	I_{15}

Tabela 8.3

O circuito que executa esta função é visto na figura 8.10.

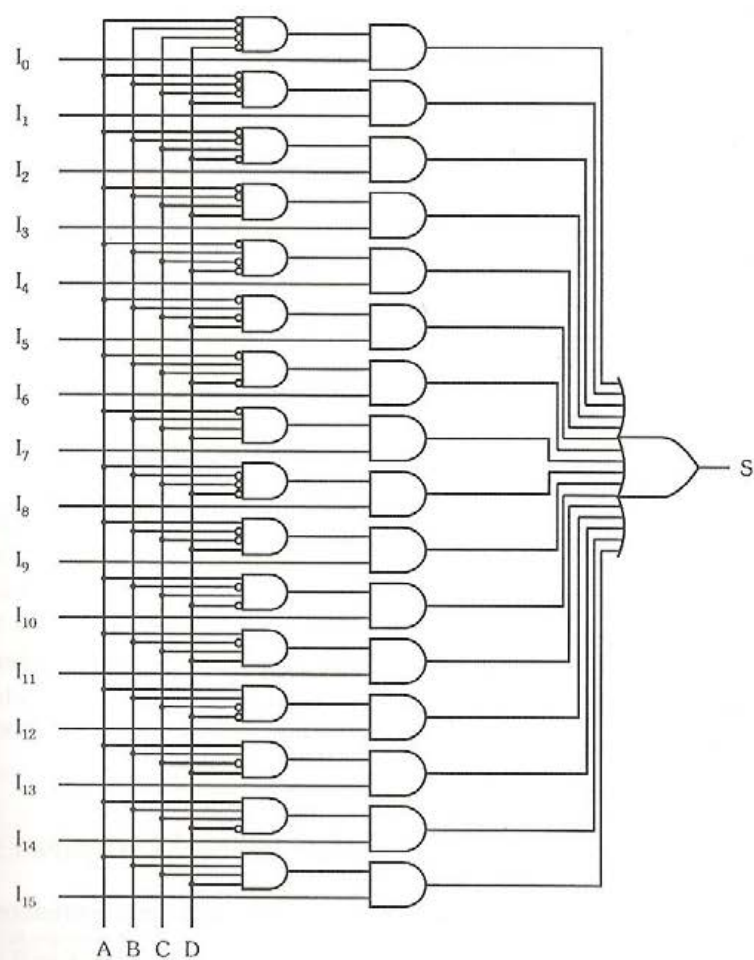



Figura 8.11

Representando apenas em bloco, temos:

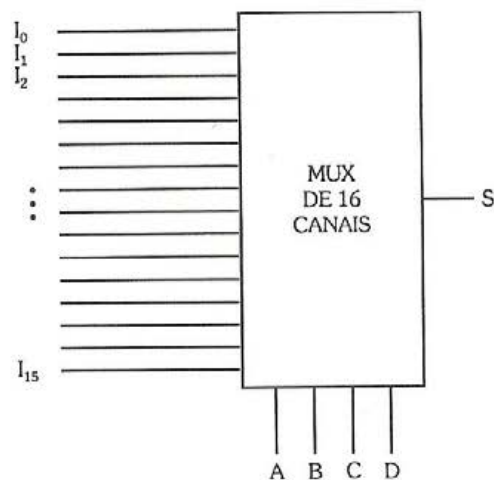



Figura 8.12

O funcionamento deste circuito é análogo ao de 2 variáveis. Podemos notar que cada informação de entrada possui apenas uma combinação das variáveis de seleção que a conecta à saída, portanto, se quisermos conectar à saída uma determinada informação, precisamos injetar nas entradas de seleção sua respectiva combinação. A essa combinação damos o nome de **endereço**, conceito facilmente compreensível, pois, ao injetarmos as variáveis de seleção, estamos endereçando através de um código binário, a informação que deve ser conectada à saída.

8.3.2 Outras Maneiras de formar um Bloco Multiplex

Podemos formar blocos multiplex através de quaisquer geradores de produtos canônicos. Esquematicamente, temos:

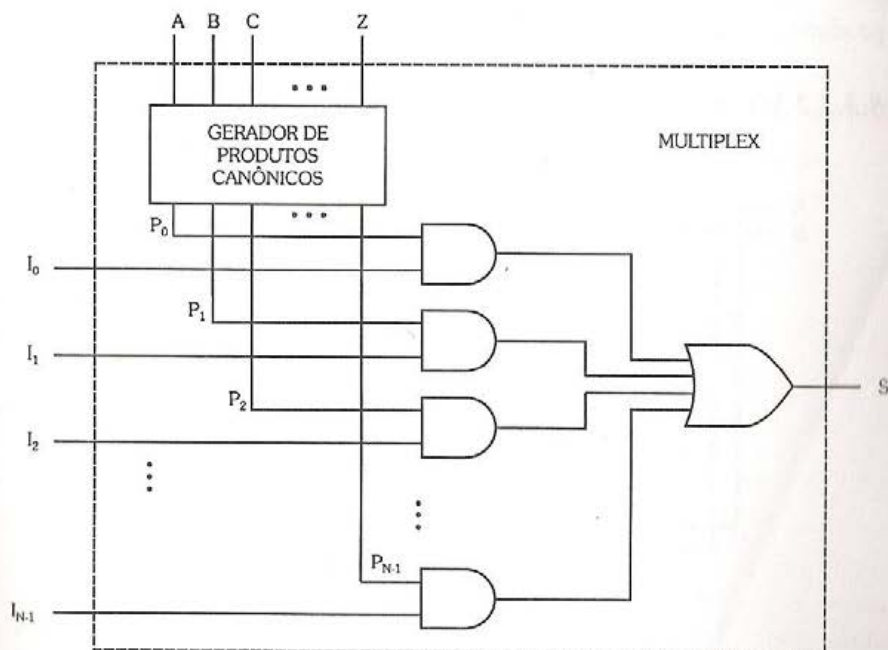



Figura 8.13

Vamos, para exemplificar, esquematizar um multiplex de 8 canais com 3 variáveis de seleção, utilizando os principais geradores de produtos canônicos.

8.3.2.1 Multiplex utilizando Matriz de Encadeamento Simples

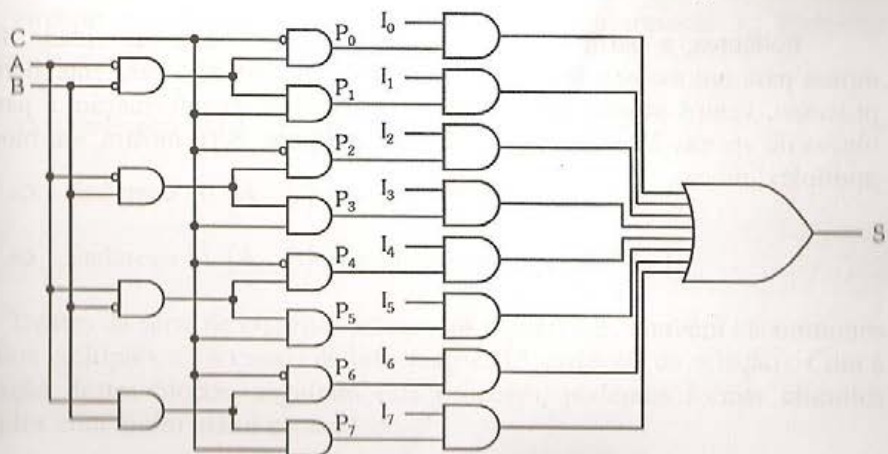



Figura 8.14

Injetando o endereço de uma dada informação, esta será desbloqueada (o produto canônico correspondente será igual a 1) e será conectada à saída.

8.3.2.2 Multiplex utilizando Matriz de Encadeamento Duplo

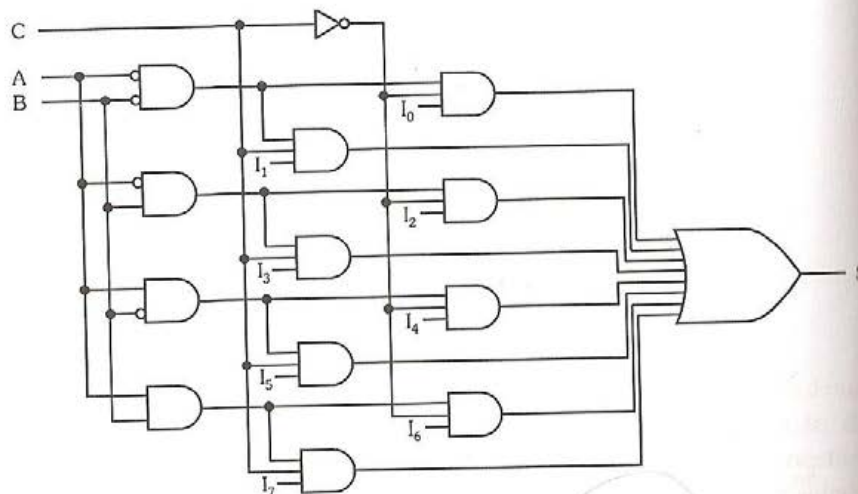



Figura 8.15

Esta maneira de construção do multiplex é uma das mais utilizadas, pois apresenta uma rápida comutação.

8.3.3 Ampliação da Capacidade de um Sistema Multiplex

Podemos, a partir de circuitos multiplex de baixa capacidade, formar outros para um maior número de informações de entrada. Para entendermos o processo, vamos montar um multiplex de 4 canais de informação, a partir de outros de apenas 2 canais de informação. A figura 8.16 mostra, em blocos, o multiplex obtido.

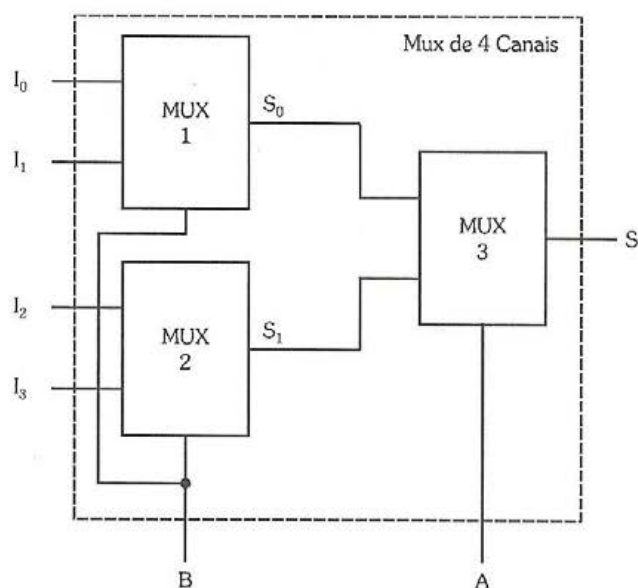



Figura 8.16

Ao entrarmos com o endereço 00 $(\overline{A} . \overline{B})$, encontramos na saída a informação I_0. Como podemos notar, no circuito, quando B for igual a 0, as saídas intermediárias S_0 e S_1 estarão com as informações, I_0 e I_2 respectivamente. Quando A for 0, teremos na saída S somente o valor de saída intermediária S_0, que neste caso estará com o valor I_0 , logo, ao injetarmos neste circuito o endereço 00, teremos na saída a informação I_0. Podemos analisar de modo análogo os outros endereços:

⇨ Endereço 01 $(\overline{A} . B)$: a saída assumirá o valor de I_1.

⇨ Endereço 10 $(A . \overline{B})$: a saída assumirá o valor de I_2.

⇨ Endereço 11 $(A . B)$: a saída assumirá o valor de I_3.

Dentro da série de circuitos integrados comerciais, também encontramos circuitos multiplex de 8 canais de informações (3 variáveis de seleção). Com a utilização destes blocos, seguindo este processo, podemos formar circuitos multiplex com muito maior capacidade.

Para ilustrar, vamos elaborar a seguir, um exemplo de confecção de circuito multiplex com capacidade superior a 8 canais. Vamos efetuar a

confecção de um multiplex de 16 canais, utilizando blocos de 8 canais de informação.

Para isso, devemos conectar os blocos da maneira vista na figura 8.17.

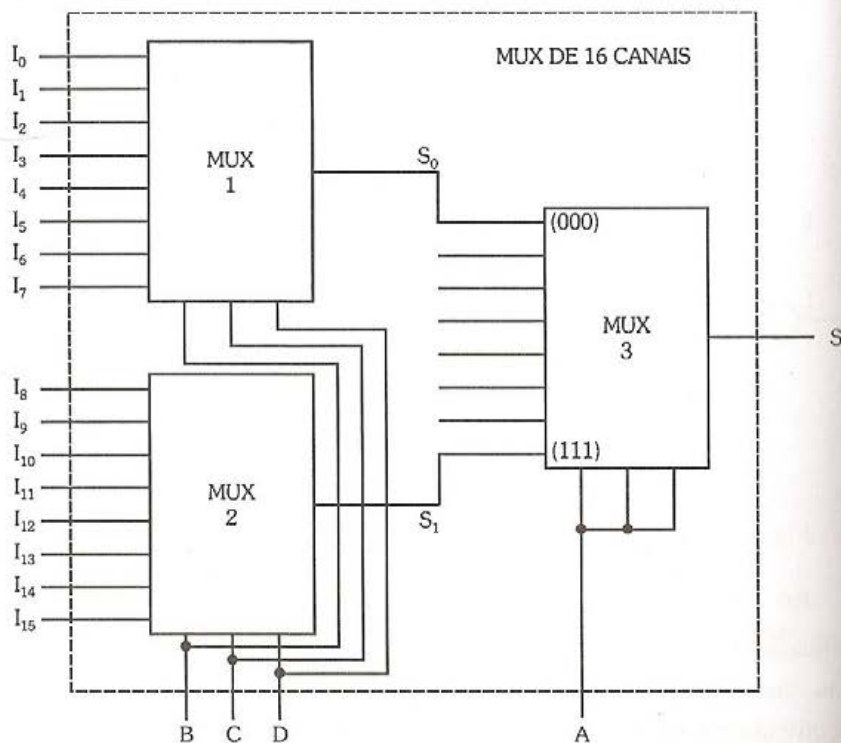



Figura 8.17

Nos blocos multiplex 1 e 2, as variáveis B, C e D irão selecionar os 2 canais de entrada, que possuem endereços iguais (BCD), nas saídas S_0 e S_1. O multiplex 3, por possuir as entradas de seleção curto-circuitadas, apresentará somente os endereços 000 (A = 0) ou 111 (A = 1), logo, este bloco efetuará a seleção final através de variável A, complementar ao endereço. Podemos observar que no multiplex 3, as saídas S_0 e S_1 deverão ser ligadas nas entradas cujos endereços são 000 e 111, pois devido ao tipo de ligação das variáveis de seleção, as outras entradas jamais serão endereçadas. Após esta análise, concluímos que o conjunto executa a função de um sistema multiplex de 16 canais de informação.

8.3.4 Endereçamento Seqüencial em um Sistema Multiplex

Podemos utilizar um multiplex que apresente, seqüencialmente na saída, os dados correspondentes aos canais de informação. Para isso, basta conectarmos às entradas de seleção um circuito contador que gere a seqüência de contagem desejada. Para ilustrar este procedimento, a figura 8.18 mostra um multiplex de 8 canais com seleção seqüencial feita por um contador de 0 a 7 (8 estados).

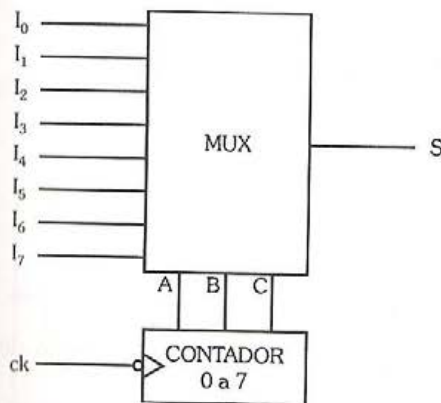



Figura 8.18

Uma das utilidades deste sistema é a conversão de uma informação paralela em uma informação série, pois se o contador gerar a seqüência binária, teremos seqüencialmente na saída, as informações I_0 , I_1, I_2 até I_{N-1}. Essa configuração, porém, não faz com que o multiplex funcione obrigatoriamente como sendo um conversor paralelo-série, pois dado o endereço de um canal de entrada, a saída irá variar de acordo com a variação deste, logo, se surgir na entrada um trem de pulsos, este será recolhido na saída.

8.3.5 Utilização do Multiplex na Construção de Circuitos Combinacionais

O circuito multiplex pode ser utilizado também para a montagem de circuitos combinacionais quaisquer. Para isso, basta montar a tabela da verdade do circuito como no capítulo 2.

As saídas que o circuito deve apresentar em cada uma das possibilidades devem ser injetadas nos canais de informação. Assim, quando ocorrer uma das

possibilidades, as variáveis de seleção irão endereçar a respectiva informação, que terá o seu valor definido de acordo com a tabela da verdade.

Para exemplificar, vamos esquematizar o circuito que executa a tabela 8.4, utilizando blocos multiplex.

A	B	C	S_1	S_2
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Tabela 8.4

Vamos, agora, estabelecer os valores dos canais de informação de cada um dos multiplex, que irão apresentar as saídas S_1 e S_2:

Variáveis de Seleção			MUX 1	MUX 2
A	B	C	S_1	S_2
0	0	0	$I_0 = 0$	$I_0 = 0$
0	0	1	$I_1 = 1$	$I_1 = 0$
0	1	0	$I_2 = 1$	$I_2 = 0$
0	1	1	$I_3 = 0$	$I_3 = 1$
1	0	0	$I_4 = 1$	$I_4 = 0$
1	0	1	$I_5 = 0$	$I_5 = 1$
1	1	0	$I_6 = 0$	$I_6 = 1$
1	1	1	$I_7 = 1$	$I_7 = 1$

Tabela 8.5

Partindo da tabela, vamos escrever os valores que as informações de entrada devem assumir:

MUX 1: $I_0 = I_3 = I_5 = I_6 = 0$

$I_1 = I_2 = I_4 = I_7 = 1$

MUX 2: $I_0 = I_1 = I_2 = I_4 = 0$

$I_3 = I_5 = I_6 = I_7 = 1$

Vamos, então, injetar esses valores nos respectivos canais de informação. O esquema do circuito, nesta situação, é visto na figura 8.19.

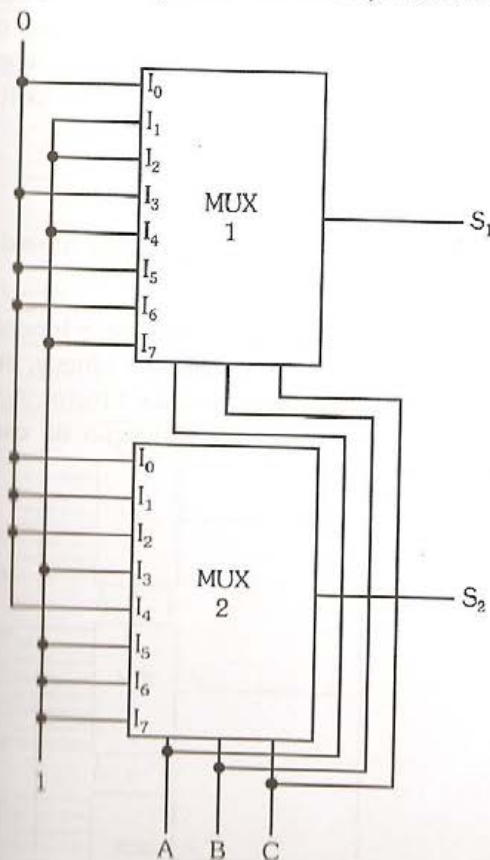



Figura 8.19

Este circuito irá apresentar as saídas S_1 e S_2 de acordo com as variáveis de seleção de entrada, seguindo a tabela da verdade.

Para verificarmos o funcionamento do circuito, vamos analisar um dos casos, pois os outros serão análogos. Analisaremos, por exemplo, o caso das entradas ABC iguais a 011, respectivamente.

Ambos os multiplex irão endereçar o canal de informação I_3, logo, nas saídas S_1 e S_2, teremos respectivamente 0 e 1, que estão colocados respectivamente nas entradas.

Este exemplo mostra que podemos esquematizar um circuito combinacional através da utilização de blocos multiplex.

A vantagem do emprego do multiplex está na facilidade de esquematização de circuitos, principalmente quando temos um número elevado de variáveis. Por exemplo, quando tivermos 8 variáveis, teremos 256 possibilidades, o que implicará numa grande dificuldade de simplificação do circuito. Utilizando este processo, basta injetarmos os valores 1 e 0 nos canais de informação, de acordo com as variáveis de seleção, conforme a tabela da verdade. Veremos mais adiante um outro processo, utilizando memórias ROM.

8.3.6 Exercícios Resolvidos

1 - Esquematize um multiplex de 64 canais, utilizando apenas blocos de 8 canais de informação.

Para obtermos um multiplex de 64 canais, necessitamos de 8 blocos de 8 canais e mais um, para efetuar a conexão final de todos os blocos, ou seja, a ligação de todos os 8 fios, das saídas pertencentes aos blocos. A figura 8.20 mostra o sistema montado e a respectiva identificação do canal de entrada inicial e final de cada bloco.

MUX DE 64 CANAIS
I_0
I_7
MUX 1
S_0
D E F
I_8
I_{15}
MUX 2
S_1
I_{16}
I_{23}
MUX 3
S_2
I_{24}
I_{31}
MUX 4
S_3
I_{32}
I_{39}
MUX 5
S_4
I_{40}
I_{47}
MUX 6
S_5
I_{48}
I_{55}
MUX 7
S_6
I_{56}
I_{63}
MUX 8
S_7
0
7
MUX 9
A B C
S_0

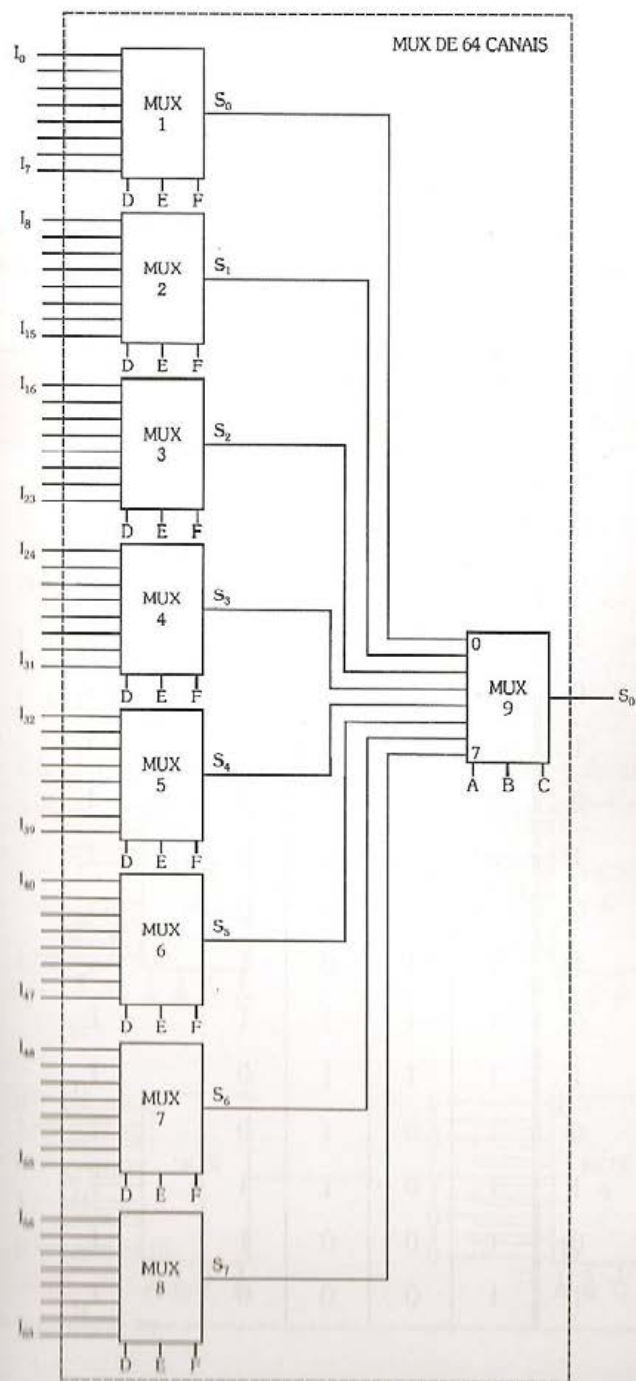



Figura 8.20

Pelo circuito, notamos que a seleção dos canais é feita pelas variáveis D, E e F em conjunto com A, B e C (64 canais ⇒ 6 fios de seleção: 2^6), sendo as três primeiras responsáveis pela seleção dos canais de entrada em cada bloco e as três últimas, pela colocação efetiva do canal na saída.

2 - Utilizando blocos multiplex, confeccione um decodificador que transforme do sistema binário comum para o código Gray.

O primeiro passo é montarmos a tabela da verdade correspondente a esta codificação, indicando conforme a seleção, os canais a serem comutados às saídas.

A	B	C	D	S_1	S_2	S_3	S_4	Canais de Informação
0	0	0	0	0	0	0	0	I_0
0	0	0	1	0	0	0	1	I_1
0	0	1	0	0	0	1	1	I_2
0	0	1	1	0	0	1	0	I_3
0	1	0	0	0	1	1	0	I_4
0	1	0	1	0	1	1	1	I_5
0	1	1	0	0	1	0	1	I_6
0	1	1	1	0	1	0	0	I_7
1	0	0	0	1	1	0	0	I_8
1	0	0	1	1	1	0	1	I_9
1	0	1	0	1	1	1	1	I_{10}
1	0	1	1	1	1	1	0	I_{11}
1	1	0	0	1	0	1	0	I_{12}
1	1	0	1	1	0	1	1	I_{13}
1	1	1	0	1	0	0	1	I_{14}
1	1	1	1	1	0	0	0	I_{15}

Tabela 8.6

No multiplex 1, temos:

$I_0 = I_1 = I_2 = I_3 = I_4 = I_5 = I_6 = I_7 = 0$

$I_8 = I_9 = I_{10} = I_{11} = I_{12} = I_{13} = I_{14} = I_{15} = 1$

No multiplex 2, temos:

$I_0 = I_1 = I_2 = I_3 = I_{12} = I_{13} = I_{14} = I_{15} = 0$

$I_4 = I_5 = I_6 = I_7 = I_8 = I_9 = I_{10} = I_{11} = 1$

No multiplex 3, temos:

$I_0 = I_1 = I_6 = I_7 = I_8 = I_9 = I_{14} = I_{15} = 0$

$I_2 = I_3 = I_4 = I_5 = I_{10} = I_{11} = I_{12} = I_{13} = 1$

No multiplex 4, temos:

$I_0 = I_3 = I_4 = I_7 = I_8 = I_{11} = I_{12} = I_{15} = 0$

$I_1 = I_2 = I_5 = I_6 = I_9 = I_{10} = I_{13} = I_{14} = 1$

Efetuando as ligações, montamos o circuito visto na figura 8.21.

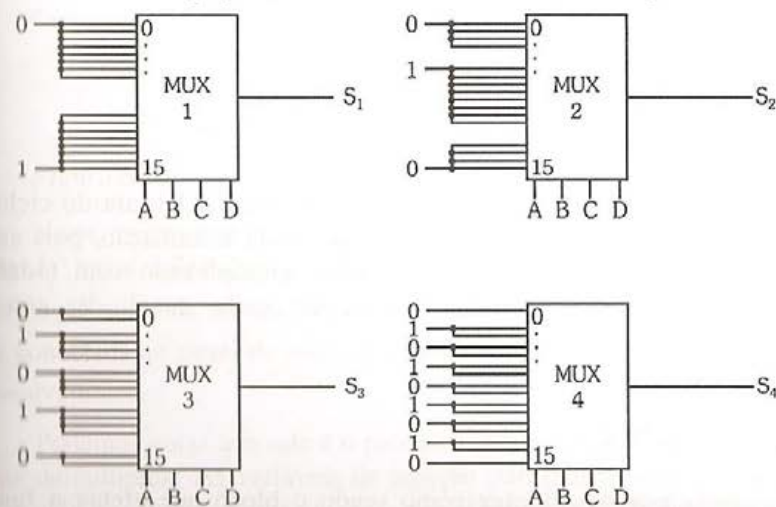



Figura 8.21

Injetando nas entradas ABCD, o código binário (A é o bit mais significativo), obtemos nas saídas S_1, S_2, S_3 e S_4, o código Gray (S_1 é o bit mais significativo).

3 - A figura 8.22 apresenta os sinais de informação de entrada e de seleção de um multiplex de 2 canais. Esboce o sinal de multiplexado.

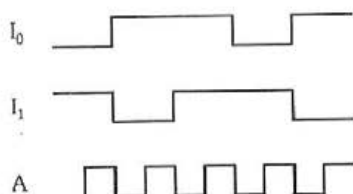



Figura 8.22

Para montamos a forma de onda da saída multiplexada, necessitamos verificar a variável de seleção (A), que estando em nível 0, seleciona o trecho relativo ao canal I_0, e em nível, o relativo ao canal I_1. Assim sendo, o sinal multiplexado é visto de modo ampliado na figura 8.23.

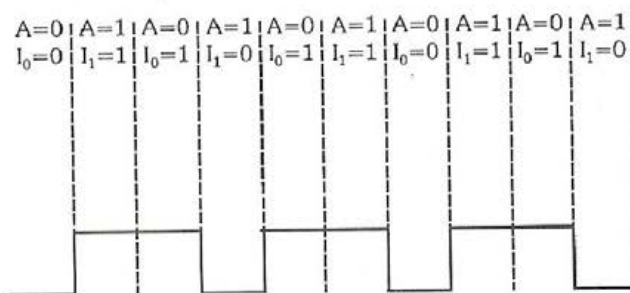



Figura 8.23

Podemos notar que para não haver perda de sinal, a largura do ciclo de seleção deve ser a metade da ocupada por cada informação, pois assim sendo, a seleção possibilita obter o sinal multiplexado com todas as amostras intercaladas da informação de cada canal de entrada, transmitindo-as completamente.

8.4 Demultiplex

Entende-se por demultiplex como sendo o bloco que efetua a função inversa ao multiplex, ou seja, a de enviar informações contidas em um canal a vários canais de saída. A figura 8.24 mostra um bloco demultiplex genérico.

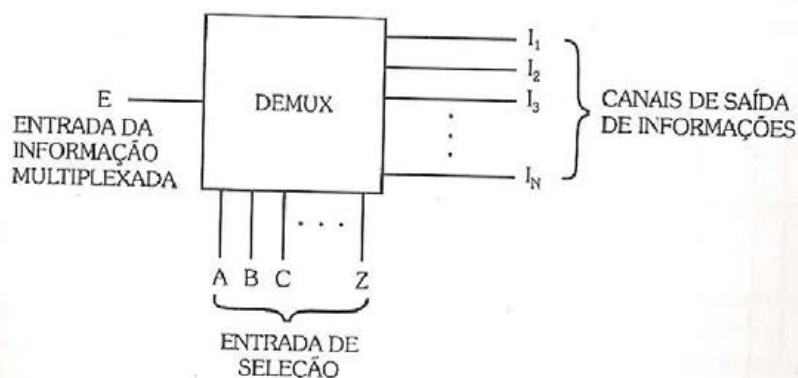



Figura 8.24

As entradas de seleção têm como finalidade escolher qual o canal de informação de saída que deve ser conectado à entrada, ou seja, devem endereçar o canal de saída, ao qual a informação deve se dirigir.

Um circuito elementar que efetua uma demultiplexação é visto na figura 8.25.

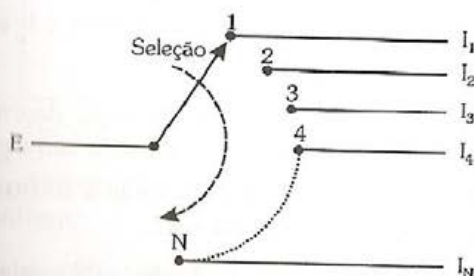



Figura 8.25

Neste circuito, se quisermos ligar a informação de entrada ao canal de saída I_1 basta selecionarmos a posição 1 da chave seletora, surgindo a informação somente na saída I_1. Se quisermos que a informação de entrada seja conectada ao canal de saída I_2, basta selecionarmos a posição 2, e assim sucessivamente.

Podemos notar que este é o processo inverso de um multiplex, vem daí o nome demultiplex. As variáveis de seleção irão indicar qual a posição que a chave seletora deve assumir, ou seja, a qual canal de saída devemos conectar a informação de entrada.

O circuito lógico básico de um demultiplex de 2 canais está esquematizado na figura 8.26.

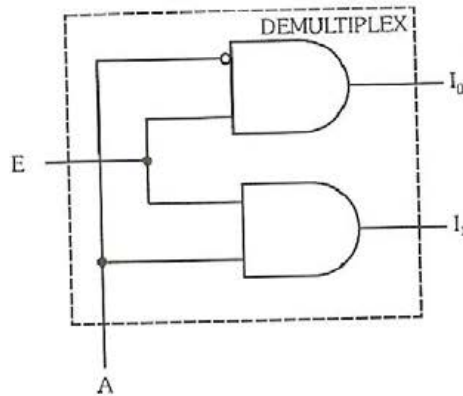



Figura 8.26

Vamos analisar o funcionamento do circuito, em função do valor assumido pela variável A:

⇨ A = 0: I_0 irá assumir o valor da entrada de informação (E), e I_1 estará em 0.

⇨ A = 1: I_1 irá assumir o valor da entrada de informação (E), e I_0 estará em 0.

Podemos notar que quando A = 0 (endereço 0), a informação de entrada saíra em I_0 e quando A = 1 (endereço 1), a informação de entrada saíra por I_1. Assim sendo, as variáveis de seleção fornecem o endereço, dado à informação de entrada, do local (canal de saída) por onde esta deverá sair.

Vamos, então, escrever estas possibilidades em uma tabela da verdade:

Variável de Seleção	Canais de Informação	
A	I_0	I_1
0	E	0
1	0	E

Tabela 8.7

8.4.1 Projeto do Circuito de um Demultiplex

Para projetarmos um demultiplex devemos relacionar, primeiramente, a possibilidade que as variáveis de seleção irão assumir (endereço), com o canal de saída de informação que deve ser conectado à entrada. Para isso, montamos uma tabela da verdade onde são consideradas todas as possibilidades de seleção e os respectivos canais de informação.

Como exemplo, vamos elaborar um demultiplex de 4 canais. Para que possamos conectar aleatoriamente uma entrada a 4 canais de saída, necessitamos, como já visto, de 2 variáveis de seleção. Com isso, podemos montar a tabela da verdade:

Variáveis		Canais de Saída			
A	B	I_0	I_1	I_2	I_3
0	0	E	0	0	0
0	1	0	E	0	0
1	0	0	0	E	0
1	1	0	0	0	E

Tabela 8.8

Através de uma tabela, notamos que, quando as variáveis de seleção assumirem:

⇨ 00 $(P_0 = \overline{A} \cdot \overline{B})$: teremos o valor de E no canal de saída I_0.

⇨ 01 $(P_1 = \overline{A} \cdot B)$: teremos o valor de E no canal de saída I_1.

⇨ 10 $(P_2 = A \cdot \overline{B})$: teremos o valor de E no canal de saída I_2.

⇨ 11 $(P_3 = A \cdot B)$: teremos o valor de E no canal de saída I_3.

O circuito para executar esta função é visto na figura 8.27.

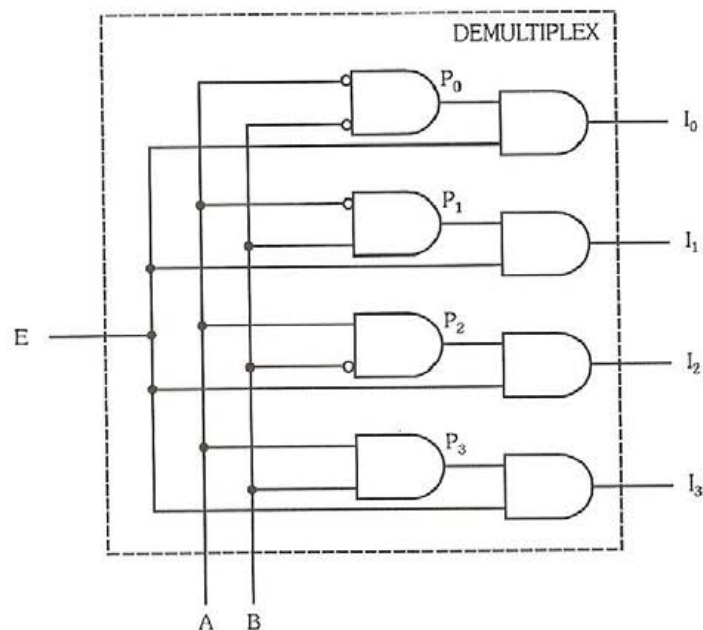



Figura 8.27

O funcionamento do circuito demultiplex é análogo ao do multiplex. Para verificarmos, vamos analisar um dos casos possíveis das variáveis de seleção, por exemplo, enviando o endereço 01 $(\overline{A} \cdot B)$.

Quando ocorrer este caso, o gerador de produtos canônicos interno ao circuito estará com P_1 em 1, com isso, a porta E ligada à saída P_1 estará com um dos terminais em nível 1, logo, em sua saída (I_1), teremos o valor ou os valores assumidos pela entrada das informações (E), permanecendo as outras saídas em nível 0.

A	B	I_0	I_1	I_2	I_3
0	1	0	E	0	0

O circuito pode também ser representado como mostra a figura 8.28.

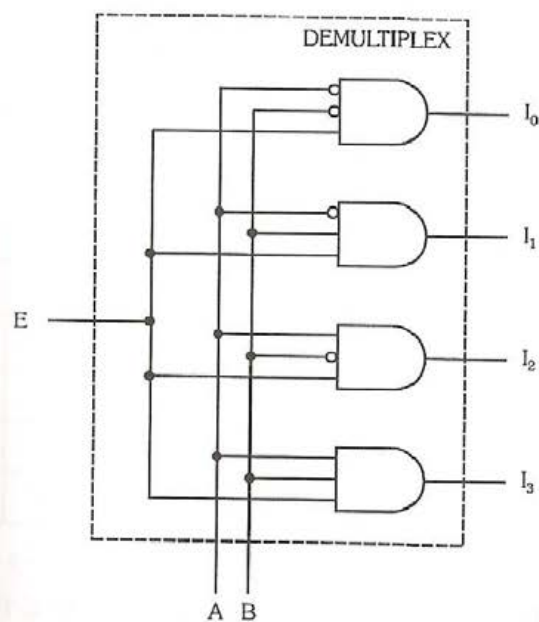



Figura 8.28

Em bloco, o circuito fica representado:

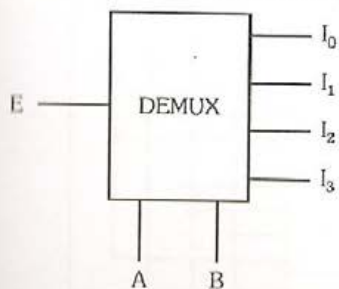



Figura 8.29

Como outro exemplo, vamos elaborar um circuito demultiplex de 8 canais de saída (3 variáveis de seleção). Seguindo o mesmo processo, vamos montar a tabela da verdade:

Variáveis de Seleção			Canais de Saída							
A	B	C	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
0	0	0	E	0	0	0	0	0	0	0
0	0	1	0	E	0	0	0	0	0	0
0	1	0	0	0	E	0	0	0	0	0
0	1	1	0	0	0	E	0	0	0	0
1	0	0	0	0.	0	0	E	0	0	0
1	0	1	0	0	0	0	0	E	0	0
1	1	0	0	0	0	0	0	0	E	0
1	1	1	0	0	0	0	0	0	0	E

Tabela 8.9

Vamos, a partir da tabela, desenhar o circuito que executa a função proposta.

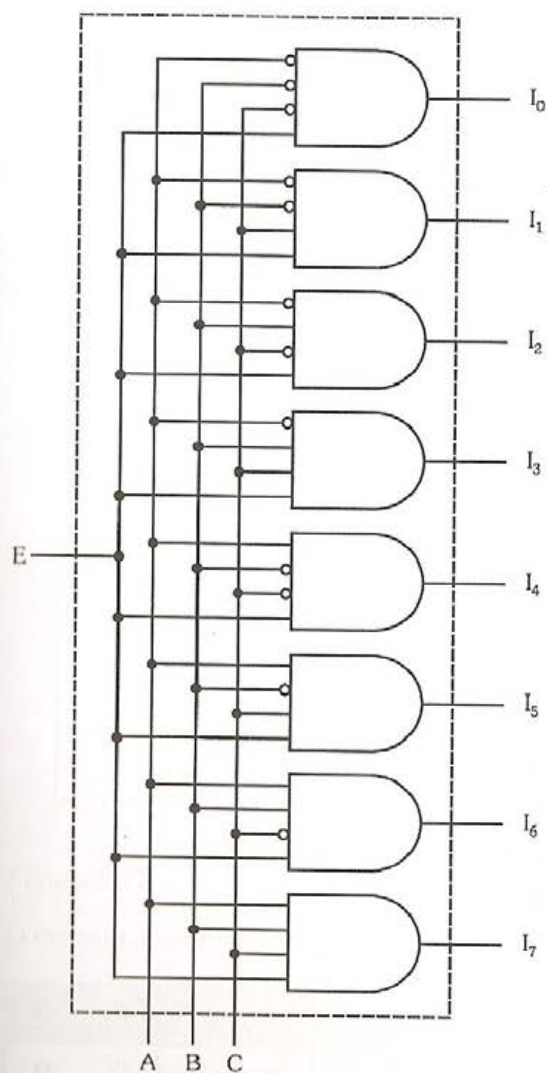



Figura 8.30

Podemos verificar que de acordo com o endereço (valores assumidos por ABC), a informação de entrada surgirá na saída respectiva. Notamos ainda, que cada canal de saída possui apenas um endereçamento.

8.4.2 Outras Maneiras de formar um Bloco Demultiplex

Podemos formar blocos demultiplexadores através de quaisquer geradores de produtos canônicos. Esquematizando, de forma geral, temos:

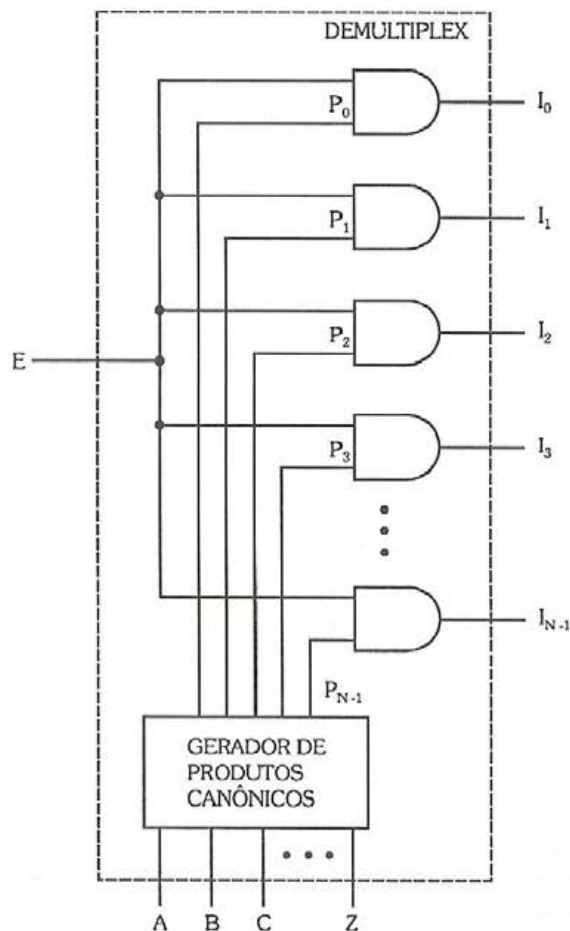



Figura 8.31

O gerador de produtos canônicos interno ao circuito funciona como distibuidor de endereços, pois de acordo com a entrada das variáveis de seleção, desbloqueará somente uma saída.

Esse gerador de produtos canônicos poderá ser construído utilizando uma matriz de encadeamento simples, matriz de encadeamento duplo ou mesmo feito através de portas E, como nos casos vistos anteriormente.

8.4.3 Ampliação da Capacidade de um Circuito Demultiplex

Como nos circuitos multiplex, podemos montar a partir de demultiplexadores de menor capacidade, outros de maior capacidade, ou seja, maior número de canais de saída.

Para entendermos o processo, vamos iniciar com um caso simples, onde vamos montar um demultiplex de 4 canais a partir de outros de apenas 2 canais de saída. A figura 8.32 apresenta esta montagem.

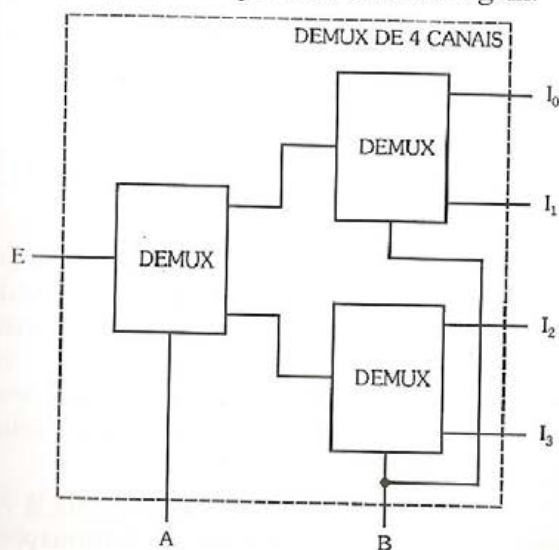



Figura 8.32

O circuito acompanhará a seguinte tabela da verdade:

A	B	S_0	S_1	S_2	S_3
0	0	E	0	0	0
0	1	0	E	0	0
1	0	0	0	E	0
1	1	0	0	0	E

Tabela 8.10

Seguindo este mesmo processo, podemos formar circuitos demultiplex de qualquer capacidade de saída.

Para ilustrar, vamos construir um demultiplex de 16 canais de saída, utilizando apenas blocos de 8 canais. O circuito é visto na figura 8.33.

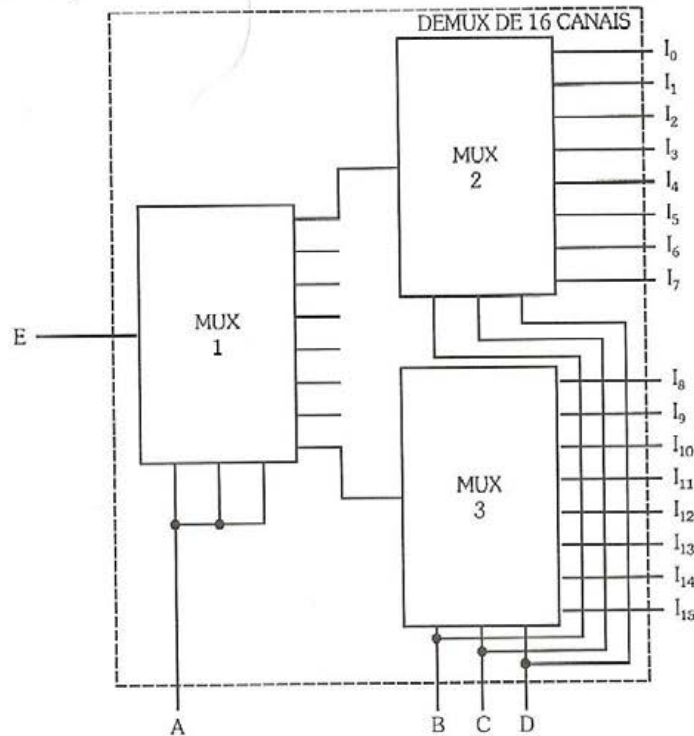



Figura 8.33

Neste caso, o demultiplex 1 receberá as entradas de informações e a primeira parte do endereço (A), com isso, selecionará através das saídas, um dos dois blocos demultiplex. A segunda parte do endereço (BCD) selecionará por qual dos canais a informação deverá sair.

8.4.4 Demultiplex com Endereçamento Seqüencial

Podemos utilizar um demultiplex que apresente a informação de entrada, saindo pelos canais de acordo com um endereçamento seqüencial. Para isso, basta conectarmos às entradas de seleção, um circuito contador que gere a contagem com a seqüência desejada. Conforme a saída do contador, a entrada de informações será conectada aos canais de saída. Desse modo, quando o contador assume estado 0, a informação sairá pelo canal de saída I_0, e quando assumir estado 1, sairá pela saída I_1, e assim sucessivamente.

A figura 8.34 apresenta a configuração de um sistema genérico montado.

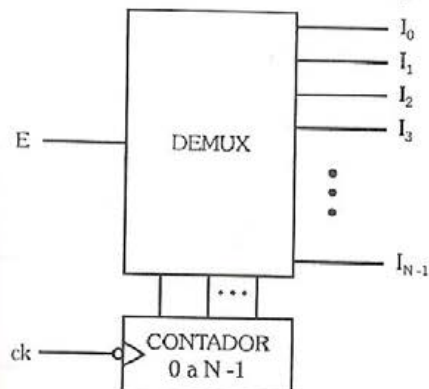



Figura 8.34

Esta configuração, da maneira como se apresenta, não permite a conversão de informação série para paralela, pois não permite a saída simultânea de informações pelos canais de saída. Um modo de solucionar o problema é o armazenamento dessas informações em flip-flops ligados às saídas e com isso recolher a informação paralela após um tempo convenientemente dimensionado, em função dos sinais de clock aplicados ao contador e aos flip-flops de saída.

8.4.5 Exercícios Resolvidos

1- Elabore um demultiplex de 8 canais, utilizando uma matriz de encadeamento simples.

O circuito deste demultiplex é semelhante ao do multiplex esquematizado no item 8.3.2.1, principalmente no que tange à geração dos produtos canônicos para o endereçamento dos canais, somente que, no caso de um demultiplex, as portas de saída irão receber a entrada E para que seja feita a demultiplexação.

Assim sendo, este circuito é visto na figura 8.35.

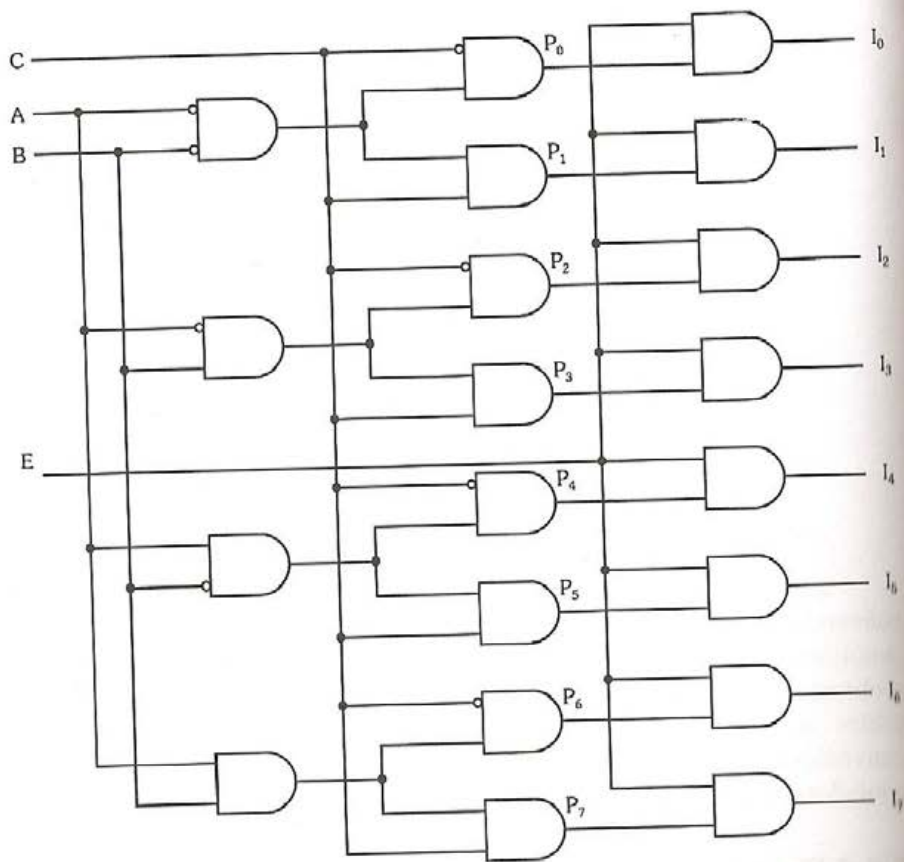



Figura 8.35

2- A figura 8.36 apresenta um demultiplex e os sinais de entrada multiplexada e de seleção. Esboce os sinais de informação.

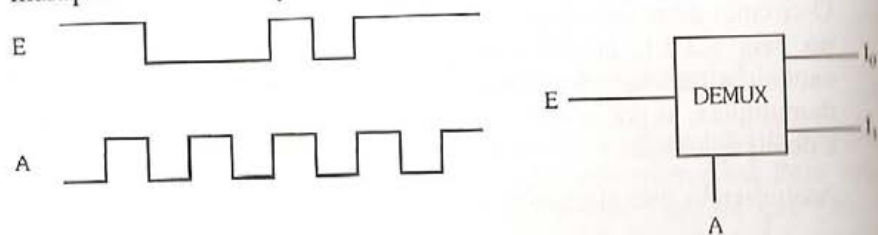



Figura 8.36

Para obtermos os sinais de informação I_0 e I_1, necessitamos verificar a variável de seleção (A), que estando em nível 0, transfere para I_0 o trecho

presente em E, e estando em nível 1, para I_1. Assim sendo, os sinais de informação obtidos, são vistos ampliados na figura 8.37.

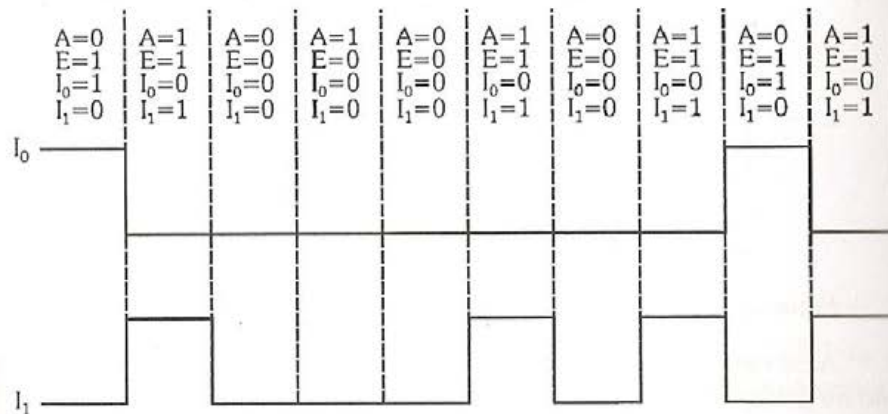



Figura 8.37

Podemos notar que quando um canal estiver sendo selecionado, o outro irá apresentar nível 0 na respectiva saída. Este fato irá ser abordado no item seguinte, relativo à transmissão de dados.

8.5 Multiplex e Demultiplex Utilizados na Transmissão de Dados

Os circuitos Multiplex e Demultiplex são muito utilizados em transmissão de dados. Para isso, basta que tenhamos um bloco no transmissor e um outro no receptor executando a função inversa. Para que haja uma perfeita recepção, é necessário também que as variáveis de seleção estejam sincronizadas, ou seja, tanto na transmissão como na recepção, as variáveis de controle devem enviar o mesmo endereço. Basicamente, temós dois processo de transmissão:

1- Transmissão paralela: através de múltiplos fios.

2- Transmissão série: através de 1 fio.

Vamos, para analisar os processos, exemplificar a transmissão de dados de 2 bits nos dois modos:

1- Transmissão Paralela

A configuração do circuito neste tipo de transmissão é vista na figura 8.38.

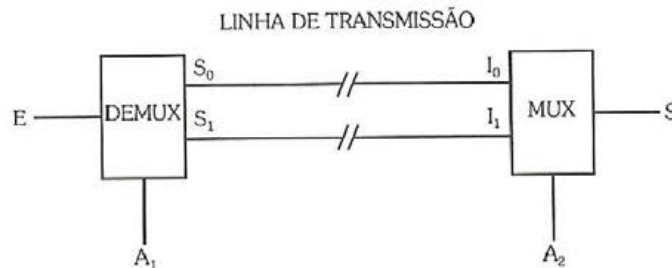



Figura 8.38

A entrada de informação E irá receber a informação de modo série, como visto no gráfico da figura 8.39.

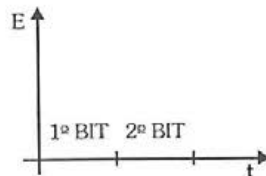



Figura 8.39

Este gráfico indica o espaço de tempo de duração do 1º e do 2º bits. Sabemos também, que os bits da informação podem assumir valores 1 ou 0.

A variável de seleção A_1 do demultiplex irá, durante o tempo de existência do 1º bit, enviar o endereço de S_0, logo este aparecerá na saída S_0. Simultaneamente, a variável de seleção A_2 do multiplex deverá enviar o mesmo endereço, fazendo com que a informação ligada em I_0, apareça na saída S. Durante a existência do 2º bit, a variável de seleção A_1 do multiplex deve enviar o endereço de S_1, logo, este aparecerá na saída S_1. Simultaneamente, a variável de seleção A_2 do multiplex deve enviar o mesmo endereço, fazendo com que a informação ligada em I_1 apareça na saída S. Assim, teremos na saída S a mesma informação aplicada à entrada E. Este processo apresenta também um caráter didático para mostrar a importância do sincronismo entre as variáveis de endereço do transmissor e do receptor, pois sem ele, a informação colhida na saída não seria verdadeira. Na prática, porém, é mais utilizado o processo visto a seguir.

2- *Transmissão Série*

A configuração do circuito é vista na figura 8.40.

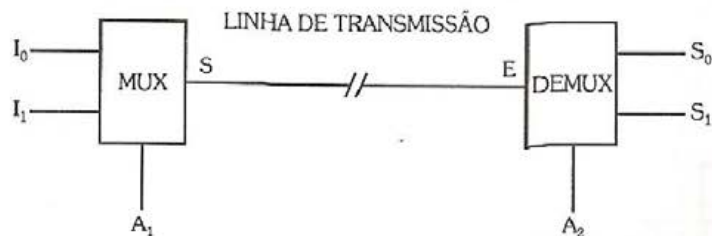



Figura 8.40

Neste caso, a entrada da informação é feita por 2 fios (2 bits de informação) e é transmitida através de um único fio. Na recepção, teremos a conversão para saída em 2 fios, como na entrada.

Para fins de análise, a figura 8.41 ilustra a entrada das informações e de seleção.

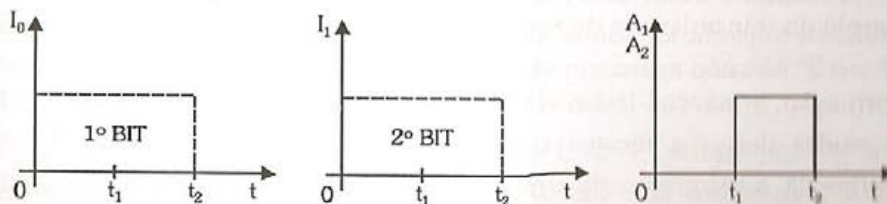



Figura 8.41

A variável de seleção A_1 do multiplex irá, durante o intervalo de tempo de 0 a t_1, enviar o endereço de I_0 (0), logo, o nível relativo ao 1º bit aparecerá na saída S, e conseqüentemente na linha de transmissão e na entrada E do bloco demultiplex de recepção. Simultaneamente, a variável de seleção A_2 do demultiplex deverá enviar o mesmo endereço, ou seja, o de S_0, fazendo com que durante esse intervalo de tempo (0 a t_1), a informação contida em S apareça em S_0. Durante o intervalo de tempo de t_1 a t_2, a variável de controle A_1 deverá enviar o endereço de I_1 (1), fazendo assim com que o nível relativo ao 2º bit apareça na saída S. Simultaneamente, a variável A_2 deverá enviar o mesmo endereço, fazendo com que, durante esse intervalo de tempo, S apareça em S_1.

A figura 8.42 mostra como a informação se comporta nos vários pontos do sistema.

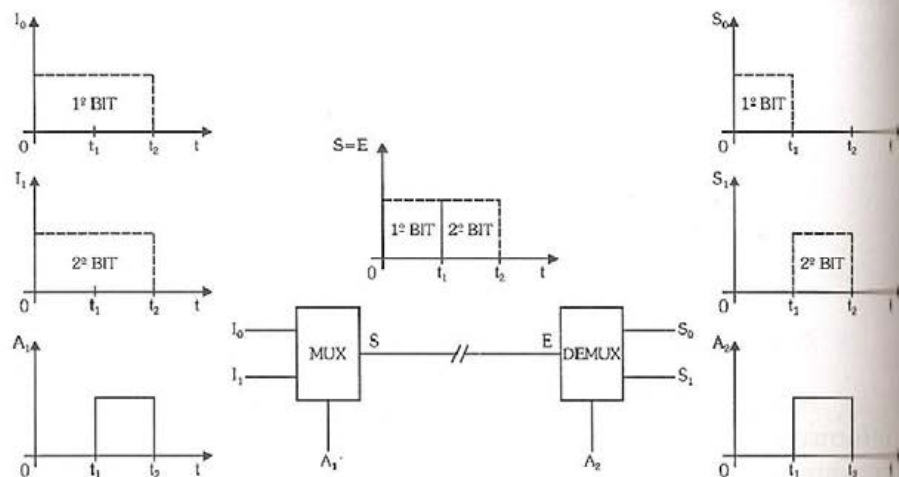



Figura 8.42

Notamos, neste caso, a importância do sincronismo das variáveis de controle do transmissor e do receptor. Notamos também, que nas saídas S_1 e S_0, o 1º e o 2º bits não aparecem simultaneamente. Podemos, então, para recolher a informação, armazená-la em flip-flops e, assim, logo após o instante t_1, termos nas saídas destes a mesma informação contida nos canais I_0 e I_1. Após o término da transmissão de uma informação, o sistema pode transmitir uma outra e, assim, transmitir várias, uma seguida à outra.

O processo apresenta a vantagem de transmitir a informação de modo série. Este fato é muito importante quando temos uma grande distância entre o transmissor e o receptor, pois a linha de transmissão poderá ser simplesmente um par de fios, linha telefônica ou, ainda, um sistema mais complexo utilizando **fibras ópticas.**

Vejamos a seguir, um sistema de transmissão de dados, utilizando multiplex e demultiplex de 8 canais de informação, ambos com endereçamento seqüencial:

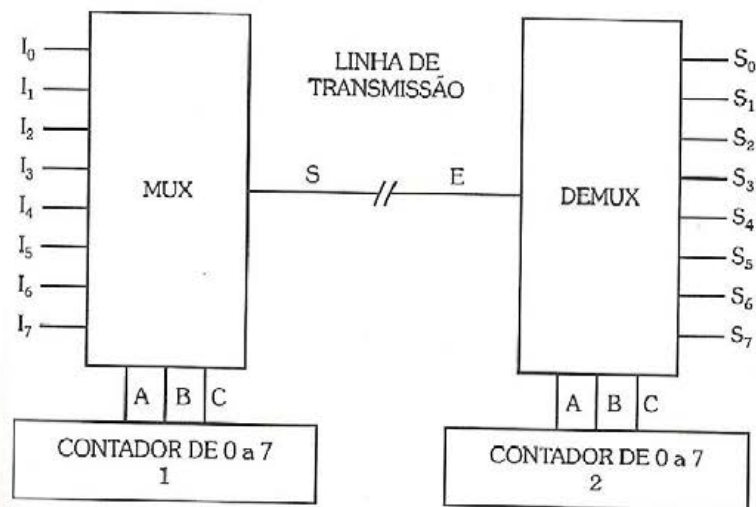



Figura 8.43

O sistema mostrado na figura 8.43, efetua a transmissão da informação, que entra através dos canais de entrada do I_0 e I_7, através de multiplexação de endereçamento seqüencial. Isso fará com que tenhamos serialmente na saída S, os bits da informação. Essa informação chegará na entrada E e será demultiplexada, também em endereçamento seqüencial.

Os bits da informação de entrada de I_0 a I_7 sairão por S_0 a S_7, respectivamente, isso se tivermos o sincronismo entre os contadores1 e 2, de transmissão e de recepção. O fato de os contadores estarem sincronizados significa que quando um deles assume um estado, o outro também assume o mesmo estado, ou seja, se o contador 1 da transmissão estiver, por exemplo, em estado 5 (endereço 101), o contador 2 da recepção também deve estar neste estado, com isso o circuito multiplex liberará o canal I_5 e o demultiplex liberará a saída S_5, logo, o bit de informação que estava no canal de entrada I_5 do multiplex sairá no canal S_5 do demultiplex.

Podemos verificar o funcionamento desse sistema através da tabela da verdade, onde, de acordo com o endereço enviado pelos contadores, relacionaremos os canais de entrada e os canais de saída.

Variáveis de Seleção			Linha	Canais de Saída							
A	B	C	S = E	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
0	0	0	I_0	I_0	0	0	0	0	0	0	0
0	0	1	I_1	0	I_1	0	0	0	0	0	0
0	1	0	I_2	0	0	I_2	0	0	0	0	0
0	1	1	I_3	0	0	0	I_3	0	0	0	0
1	0	0	I_4	0	0	0	0	I_4	0	0	0
1	0	1	I_5	0	0	0	0	0	I_5	0	0
1	1	0	I_6	0	0	0	0	0	0	I_6	0
1	1	1	I_7	0	0	0	0	0	0	0	I_7
multiplex				demultiplex							

Tabela 8.11

A figura 8.44 mostra os gráficos da linha de transmissão e das saídas para a verificação do comportamento das informações nos vários pontos do sistema.

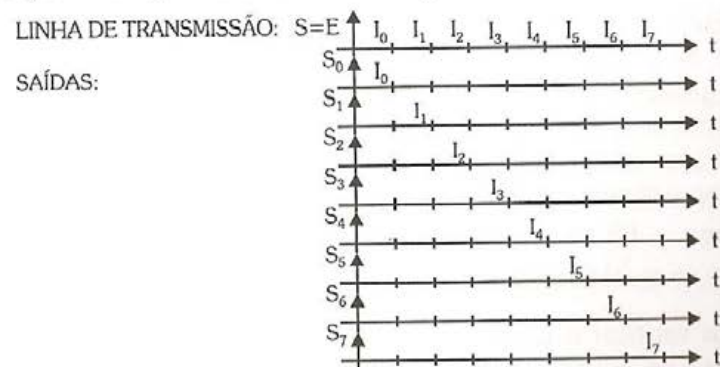



Figura 8.44

Para que tenhamos os bits de informação simultâneos, necessitamos armazená-los em flip-flops e efetuar a leitura somente ao término da transmissão completa, podendo, logo após, iniciar a transmissão de outra informação.

8.5.1 Gerador de Paridade

Normalmente, em transmissão de dados, é comum enviarmos um bit a mais na informação. Este bit, denominado **bit de paridade**, levará a seguinte informação:

1 → se foi transmitido na informação um número par de bits iguais a 1.

0 → se foi transmitido na informação um número ímpar de bits iguais a 1.

No receptor, uma vez recebida a informação mais o bit de paridade, um outro sistema irá conferir se a informação foi recebida corretamente, ou seja, se foi enviado um número par de bits iguais a 1, sendo bit de paridade igual a 1, ou enviado um número ímpar de bits igual a 1, sendo o bit de paridade igual a 0. Este sistema deve indicar se a informação foi recebida corretamente, caso contrário, deve indicar ao receptor a rejeição da mesma, pois a informação recebida não é verdadeira.

Vamos, primeiramente, estudar o circuito que gera o bit de paridade. Este circuito deve fornecer em sua saída, 1 se o número de bits iguais a 1 for par, e 0 quando este número for ímpar.

Podemos, agora, estabelecida a função do gerador de paridade, levantar sua tabela da verdade. Vamos supor que a informação a ser transmitida contenha 4 bits:

I_3	I_2	I_1	I_0	P	
0	0	0	0	1	(0 bits = 1 → P = 1)
0	0	0	1	0	(1 bit = 1 → P = 0)
0	0	1	0	0	(1 bit = 1 → P = 0)
0	0	1	1	1	(2 bits = 1 → P = 1)
0	1	0	0	0	.
0	1	0	1	1	.
0	1	1	0	1	.
0	1	1	1	0	
1	0	0	0	0	
1	0	0	1	1	

Tabela 8.12 (parte)

I_3	I_2	I_1	I_0	P
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Tabela 8.12

Vamos simplificar e esquematizar o circuito que executa esta função:

	$\overline{I_1}$		I_1		
	1	0	1	0	$\overline{I_2}$
$\overline{I_3}$	0	1	0	1	I_2
I_3	1	0	1	0	I_2
	0	1	0	1	$\overline{I_1}$
	$\overline{I_0}$	I_0	I_0	$\overline{I_0}$	

Figura 8.45

Do mapa, obtemos a expressão simplificada:

$S = \overline{I_0 \oplus I_1 \oplus I_2 \oplus I_3}$

O circuito a partir desta, é visto na figura 8.46.

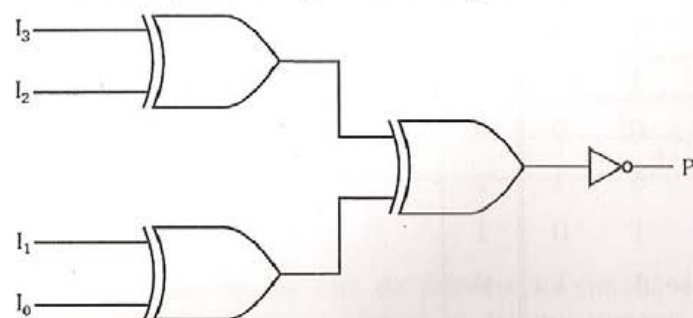



Figura 8.46

Vamos analisar o comportamento deste circuito na transmissão de um dado. Vamos supor que o dado de informação de 4 bits seja transmitido em 4 linhas, ou seja, em paralelo. A conexão do gerador de paridade é feita como mostrado na figura 8.47.

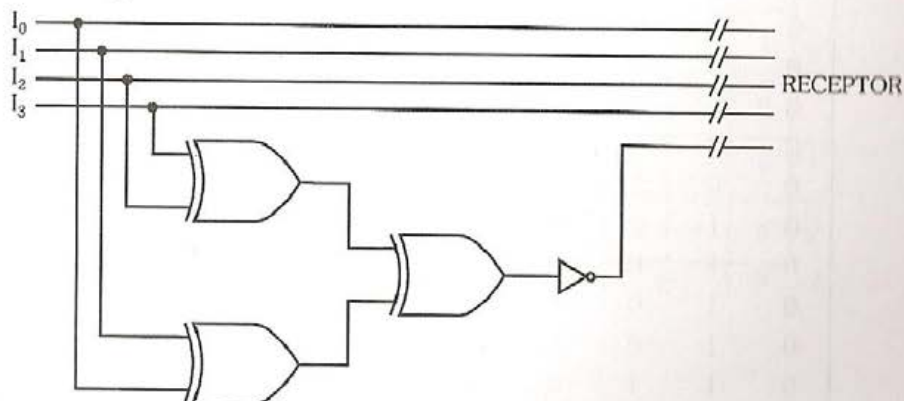



Figura 8.47

O quinto fio da linha de transmissão, ou seja, o quinto bit da informação enviada será o bit de paridade.

Devemos agora, elaborar um circuito que na recepção do dado, efetue o teste de verificação da paridade do dado transmitido.

Na recepção, teremos a informação recebida e também um bit que informará se houve ou não a paridade do número de bits desta informação. Se esta informação recebida possuir um número par de bits iguais a 1 e o bit de paridade for igual a 1; ou se possuir um número ímpar de bits iguais a 1, e o bit de paridade for igual a 0, significa que a informação recebida é correta, se, no entanto, ocorrer algo diferente disso, significa que a informação recebida não é correta. O circuito verificador de paridade deve apresentar saída 0, quando a informação recebida for correta, caso contrário deve apresentar saída igual a 1.

Com isso, podemos levantar a tabela da verdade referente a esse circuito. Nesta tabela, as variáveis serão os bits da informação recebida, incluindo também o bit de paridade:

I_3	I_2	I_1	I_0	P	S
0	0	0	0	0	1
0	0	0	0	1	0
0	0	0	1	0	0
0	0	0	1	1	1
0	0	1	0	0	0
0	0	1	0	1	1
0	0	1	1	0	1
0	0	1	1	1	0
0	1	0	0	0	0
0	1	0	0	1	1
0	1	0	1	0	1
0	1	0	1	1	0
0	1	1	0	0	1
0	1	1	0	1	0
0	1	1	1	0	0
0	1	1	1	1	1
1	0	0	0	0	0
1	0	0	0	1	1
1	0	0	1	0	1
1	0	0	1	1	0
1	0	1	0	0	1
1	0	1	0	1	0
1	0	1	1	0	0
1	0	1	1	1	1
1	1	0	0	0	1
1	1	0	0	1	0
1	1	0	1	0	0
1	1	0	1	1	1
1	1	1	0	0	0
1	1	1	0	1	1
1	1	1	1	0	1
1	1	1	1	1	0

Tabela 8.13

Vamos, agora, simplificar e esquematizar o circuito:

$\overline{I_3}$

	$\overline{I_0}$	$\overline{I_0}$	I_0	I_0	
	1	0	1	0	$\overline{I_1}$
$\overline{I_2}$	0	1	0	1	I_1
I_2	1	0	1	0	I_1
	0	1	0	1	$\overline{I_1}$
	$\overline{P}$	P	P	$\overline{P}$	

I_3

	$\overline{I_0}$	$\overline{I_0}$	I_0	I_0	
	0	1	0	1	$\overline{I_1}$
$\overline{I_2}$	1	0	1	0	I_1
I_2	0	1	0	1	I_1
	1	0	1	0	$\overline{I_1}$
	$\overline{P}$	P	P	$\overline{P}$	

Figura 8.48

$S = \overline{I_0 \oplus I_1 \oplus I_2 \oplus I_3 \oplus P}$

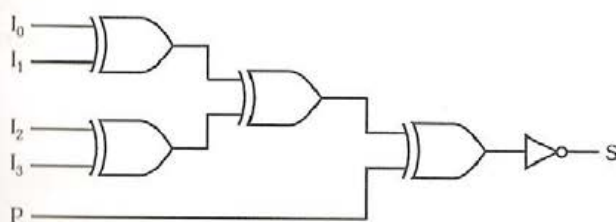



Figura 8.49

Vamos analisar o comportamento de ambos os circuitos na transmissão e recepção de um dado. Vamos supor que os dados de informação de 4 bits sejam transmitidos em 4 linhas, ou seja, em paralelo. A conexão de ambos os circuitos é mostrada na figura 8.50.

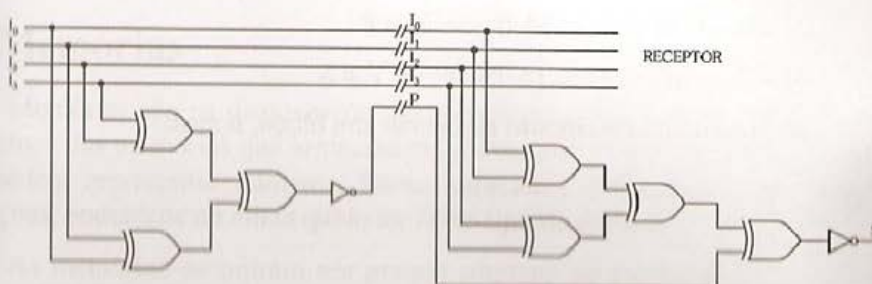



Figura 8.50

De acordo com a informação (I_0 I_1 I_2 I_3), na saída do transmissor, o circuito gerador de paridade enviará um bit, informando se o número de bits iguais a 1 é par (1) ou ímpar (0). Na recepção, o circuito verificador de paridade irá comparar o número de bits iguais a 1; da informação (I_0 I_1 I_2 I_3) com o bit de paridade enviado. De acordo com o funcionamento do circuito verificador de paridade, a saída S indicará na recepção se a informação recebida é verdadeira (S = 0) ou se é falsa, (S = 1).

Podemos também esquematizar um circuito que funcione como gerador ou verificador de paridade. A vantagem do circuito é que o mesmo bloco pode executar tanto uma como a outra função.

Este circuito nada mais é que uma extensão do circuito gerador de paridade, pois irá comparar a saída P deste com uma variável auxiliar X através de um OU Exclusivo. Se X for igual a 0, P será comparado através de um OU Exclusivo com 0, logo, se for igual a 1, a saída será 1, se for igual a 0, a saída será 0, portanto, se X for igual a 0, este circuito funcionará como um gerador de paridade.

Se na recepção aplicarmos na entrada X o bit de paridade recebido, o circuito funcionará como verificador de paridade, pois irá comparar através da porta OU Exclusivo, o bit de paridade recebido e o gerado a partir da informação recebida.

O circuito, para tal aplicação, é visto na figura 8.51.

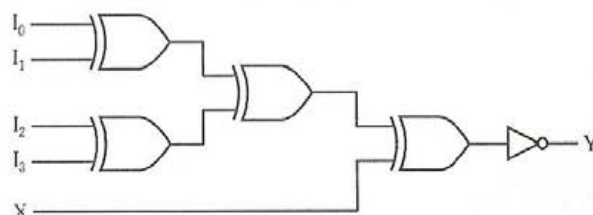



Figura 8.51

X = 0: gerador de paridade → Y = P

X = P: verificador de paridade → Y = S

Representando o circuito através de um bloco, temos:

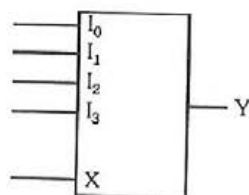



Figura 8.52

A figura 8.53 mostra um sistema completo de transmissão e recepção de dados, utilizando multiplex, demultiplex e gerador/verificador do bit de paridade.

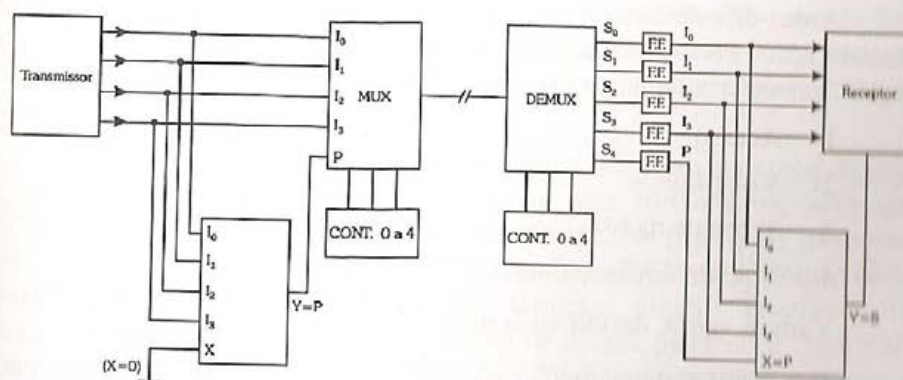



Figura 8.53

Para o sistema proposto funcionar corretamente, é necessário que os contadores atuem de maneira sincronizada e, ainda, que os flip-flops nas saídas do demultiplex, através da devida sincronização dos pulsos de clock, armazenem gradativamente a informação recebida para ser entregue ao receptor e ao verificador de paridade. Além do bit de paridade, na prática, são transmitidos também outros bits para toda a sincronização do sistema.

8.6 Memórias

Memórias são os dispositivos que armazenam informações. Neste item, trataremos das memórias que armazenam informações codificadas digitalmente que podem representar números, letras, caracteres quaisquer, comandos de operações, endereços ou ainda qualquer outro tipo de dado.

As memórias encontram seu grande emprego no campo da informática, sendo utilizadas principalmente em computadores e periféricos. São também utilizadas em outros sistemas com microprocessadores, tais como: kits e

projetos específicos. Armazenam dados para endereçamento, programação e para constituir o conjunto de programas internos para funcionalidade do próprio sistema. Um outro tipo de aplicação consiste em utilizá-las para executarem quaiquer funções de circuitos combinacionais, e ainda, com o auxílio de contadores comuns e conversores, gerar formas de onda de diversas maneiras de modo mais simples.

Nos itens seguintes, abordaremos os conceitos preliminares e itens relativos à classificação das memórias.

8.6.1 Classificação das Memórias

Antes de estudarmos os diversos tipos de memórias, vamos conhecer sua classificação. Podemos classificar as memórias em vários itens diferentes. A seguir, vamos relacionar os principais:

1- Acesso

2- Volatilidade

3- Troca de dados

4- Tipo de armazenamento

Vamos, agora, definir cada item:

1- Acesso

As memórias acessam informações em lugares denominados **localidades de memória**. Cada uma das localidades de memória possui um conjunto de bits que nos permite o seu acesso. A este conjunto de bits damos o nome de endereço. Esse conceito é de fácil compreensão, pois como o próprio nome diz, o conjunto de bits representa o endereço da localidade onde está armazenada uma informação.

O **tempo de acesso** de uma memória é o tempo necessário desde a entrada de um endereço até o momento em que a informação apareça na saída. Para as memórias de escrita/leitura é também o tempo necessário para a informação ser gravada.

Podemos ter acesso a uma dada localidade de memória de duas maneiras diferentes:

⇨ **acesso seqüencial**

⇨ **acesso aleatório**

As memórias que utilizam o acesso seqüencial, dado o endereço de uma certa localidade, permitem que se chegue até esta, passando por todas as

localidades intermediárias. As memórias mais comuns com este tipo de acesso são as que operam com fitas magnéticas, sendo utilizadas como memória de massa em computadores (para grande quantidade de dados).

Para entendermos melhor o acesso seqüencial, tomemos o exemplo de uma fita magnética. Para que tenhamos acesso a uma informação armazenada em uma localidade qualquer, necessitamos enrolar a fita até o ponto dessa localidade, para só, então, termos acesso à informação lá contida. Notamos, neste caso, que ao enrolarmos a fita, passamos por todas as localidades intermediárias.

Uma característica importante deste tipo de acesso é que o tempo de acesso depende do lugar onde a informação está armazenada. No caso da fita, se uma informação estiver no fim do rolo, necessitamos enrolá-la até o ponto desejado, logo o tempo de acesso será longo. Caso a informação esteja no início da fita, o tempo de acesso será menor.

As memórias que utilizam o acesso aleatório, dado um endereço de uma certa localidade, permitem que se chegue até esta diretamente, sem que necessitemos passar pelas localidades intermediárias. As principais memórias com este tipo de acesso são também conhecidas como **RAM (Random-Access Memory)**. São largamente utilizadas em sistemas digitais programáveis. Possuem a grande vantagem de ter um tempo de acesso pequeno e igual para qualquer uma das localidades de memória. Analisaremos mais adiante o circuito da memória RAM.

2- *Volatilidade*

Quanto à volatilidade, as memórias podem ser **voláteis** ou **não-voláteis**.

As memórias voláteis são aquelas que, ao ser cortada a alimentação, perdem as informações armazenadas. São memórias feitas, geralmente, a partir de semicondutores e na maioria das vezes, possuem como elemento de memória o flip-flop. Um exemplo típico já citado, é o da memória RAM.

As memórias não voláteis são aquelas que mesmo sem alimentação, continuam com as informações armazenadas. Dentre essas se destacam as memórias magnéticas e as eletrônicas: **ROM, PROM** e **EPROM.**

3- *Troca de Dados*

No que se refere à troca de dados com outros componentes do sistema, as memórias podem ser de escrita/leitura ou memórias apenas de leitura.

As memórias de escrita/leitura são aquelas que permitem acesso a uma localidade qualquer para armazenar a informação desejada, além disso,

permitem o acesso também para a leitura do dado. As memórias RAM também se enquadraram nesta situação.

As memórias apenas de leitura, como o próprio nome diz, são aquelas em que a informação é fixa, só podendo efetuar-se a leitura. São também conhecidas como **ROM (Read-Only Memory)**. A análise deste tipo de memória será feita adiante.

4- Tipos de Armazenamento

Quanto ao tipo de armazenamento, as memórias classificam-se em **estáticas e dinâmicas.**

As memórias de armazenamento estático são aquelas em que uma vez inserido o dado numa dada localidade, este lá permanece.

As memórias de armazenamento dinâmico são aquelas em que necessitamos inserir a informação de tempos em tempos, pois de acordo com as características de seus elementos internos, perdem essas informações após um determinado tempo.

As memórias de armazenamento estático apresentam a vantagem de possuir uma utilização de maneira mais fácil que as dinâmicas.

8.6.2 Estrutura Geral e Organização de uma Memória

Como vimos, uma memória armazena ou acessa as informações digitais mediante endereçamento, em lugares denominados localidades de memórias. Para o acesso a estas localidades o bloco possui uma série de terminais de entrada de endereços que são ligados a um conjunto de fios denominado **barra de endereços (address bus**, em inglês), sendo este responsável por todo o endereçamento de um sistema típico com microprocessador. Para a entrada e saída dos dados , da mesma forma , o bloco possui uma série de terminais ligados à **barra de dados (data bus)**. Além disso, o bloco possui terminais de controle ligados à **barra de controle (control bus)**. A figura 8.54 apresenta a esquematização de uma memória eletrônica típica com a ligação dos **barramentos** mencionados e mais os terminais de alimentação e terra.

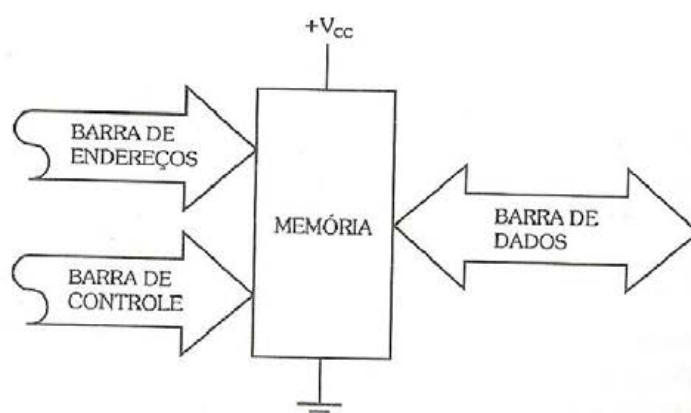



Figura 8.54

A simbologia utilizada na figura, mostra que a barra de dados é **bidirecional**, ou seja, pode ser usada tanto como saída ou entrada de dados, isso nos tipos mais comuns de memórias de escrita/leitura (RAM), sendo, um terminal apropriado da barra de controle responsável por este procedimento, a ser estudado detalhadamente mais adiante.

As memórias, de maneira geral, no que se refere à quantidade de dados armazenados, são especificadas pela notação **Nxm**, onde a primeira letra indica o número de localidades da memória e a segunda indica o número de bits da informação armazenada por localidade. Estas especificações caracterizam toda a organização de estrutura da memória. Para exemplificar, vamos relacionar algumas estruturas de memórias usuais na prática:

- 32x8
- 128x8
- 1Kx4
- 64Kx8
- 2Mx16

Notamos, por estas especificações, que o número de localidades é sempre múltiplo de 2^n, fato derivado da possibilidade total de endereçamento por um determinado número de fios ou terminais (n), em situação binária.

A designação **K (Kilo)** que significa um fator $2^{10} = 1024$, e a **M (Mega)** que significa $2^{20} = 1048576$ são muito usuais na atualidade, principalmente esta última, devido à grande quantidade de memória exigida pelos sistemas digitais. Por exemplo, a memória mencionada anteriormente de 64Kx8 possui 64x1024

= 65536 localidades, com 8 bits (1 byte) em cada uma, necessitando de 16 terminais para endereçamento. A de 2Mx16 possui 2x1048576 = 2097152 localidades com 16 bits, necessitando de 21 terminais para endereçamento.

Outro parâmetro a ser definido é o da **capacidade de memória**, que significa o número total de bits que podem ser armazenados em uma memória. Para seu cálculo, basta efetuarmos o produto Nxm , multiplicando o número de localidades pelo número de bits por localidade, obtendo assim a capacidade total em bits desta memória. Por exemplo, uma memória de 1Kx4 possui na totalidade 4096 bits de capacidade.

Um outro ponto importante a ser abordado é em relação à **palavra de endereço**, que é definida como sendo o conjunto de níveis lógicos ou bits necessários para o endereçamento de uma determinada localidade de memória para o acesso ao dado. Para facilitar a escrita da palavra de endereço relativa a cada localidade, bem como sua utilização em programação, é comum transcrever-se este conjunto de bits diretamente para hexadecimal, principalmente no caso de memórias de alta capacidade, pois, conforme já visto, este sistema de numeração permite a representação de cada 4 bits utilizando apenas 1 dígito hexadecimal através de conversão direta.

Toda a estrutura com os endereços e dados armazenados é freqüentemente colocada em uma tabela denominada **mapeamento de memória.** Para exemplificar, a tabela 8.14 apresenta o mapeamento de uma memória genérica de 256 localidades.

Endereço das Localidades em binário	Endereço das Localidades em Hexadecimal	Localidade	Conteúdo
A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0			
0 0 0 0 0 0 0 0	00	L_0	I_0
0 0 0 0 0 0 0 1	01	L_1	I_1
0 0 0 0 0 0 1 0	02	L_2	I_2
0 0 0 0 0 0 1 1	03	L_3	I_3
.	.	.	.
.	.	.	.
.	.	.	.

Tabela 8.14 (parte)

Endereço das Localidades em binário	Endereço das Localidades em Hexadecimal	Localidade	Conteúdo
1 0 1 0 0 1 1 1	A7	L167	I_{167}
1 0 1 0 1 0 0 0	A8	L168	I_{168}
1 0 1 0 1 0 0 1	A9	L169	I_{169}
.	.	.	.
.	.	.	.
.	.	.	.
1 1 1 1 1 1 0 1	FD	L253	I_{253}
1 1 1 1 1 1 1 0	FE	L254	I_{254}
1 1 1 1 1 1 1 1	FF	L255	I_{255}

Tabela 8.14

Notamos que uma memória com 256 localidades precisa de 8 fios para endereçamento ($2^8 = 256$), identificados de A_7 até A_0, sendo o endereço da localidade inicial 00_{16} (00000000_2) e da final FF_{16} (11111111_2). Supondo que a referida memória possua 8 bits por localidade, ou seja, 256x8, sua esquematização em bloco, com a barra de dados identificada de D_7 até D_0, é mostrada na figura 8.55.

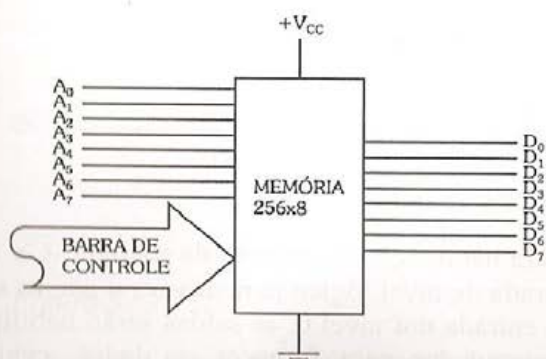



Figura 8.55

Nos itens subsequentes, vamos estudar os principais tipos de memórias e suas respectivas arquiteturas.

8.6.3 Memórias ROM

As memórias ROM, conforme já visto no item relativo à classificação, apresentam como característica principal, permitir somente a leitura dos dados nela gravados previamente em sua fabricação. Vem daí o nome ROM (Read-Only Memory), que significa memória apenas de leitura. Além disso, possuem acesso aleatório e são não voláteis, pois não perdem seus dados armazenados com o desligamento da alimentação. Na realidade, as memórias ROM podem ser consideradas como circuitos combinacionais, pois apresentam as saídas de dados em função das combinações entre as variáveis de entrada (endereçamento).

Dentre as diversas aplicações, destacamos sua utilização para o armazenamento de programas de sistemas operacionais em computadores e outros sistemas digitais. Podem, ainda, ser utilizadas em circuitos de geração de caracteres e para a construção de um circuito combinacional qualquer. A figura 8.56 apresenta o bloco representativo de uma memória ROM, com terminais e barramentos conhecidos e mais um terminal de controle ($\overline{CS}$), para **habilitação** da **pastilha** ou **chip** (em inglês).

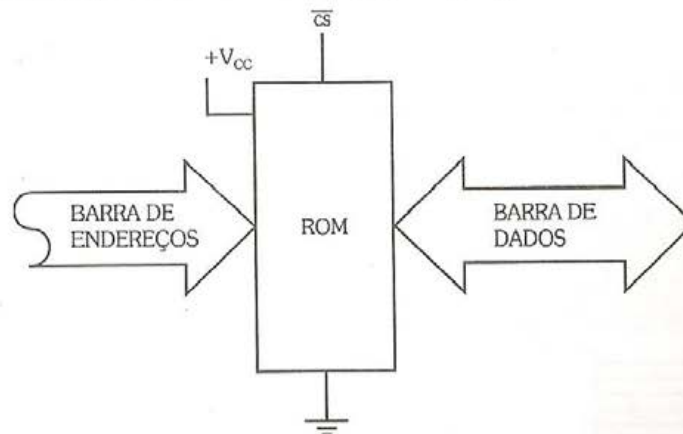



Figura 8.56

O terminal de controle para habilitação ou seleção da pastilha, $\overline{CS}$ (**chip select**) é, na realidade, uma entrada de nível lógico para ativar ou não as saídas da ROM. Se aplicarmos a esta entrada um nível 0, as saídas serão habilitadas, ou seja, serão internamente comutadas para fornecer os dados, conforme funcionamento normal de endereçamento, porém, se aplicarmos um nível 1, estas serão desabilitadas, assumindo estados de alta impedância (**tri-state**, ver capítulo sobre "Famílias Lógicas"), liberando a barra de dados para utilização

por outros dispositivos presentes no sistema controlado normalmente por microprocessador. O traço sobre CS ($\overline{CS}$), indica que a habilitação da pastilha é feita com nível 0 (ativa em 0), sendo esta uma forma de nomenclatura muito utilizada na prática. Na série de circuitos integrados comerciais, é também encontrado o termo **chip enable** ($\overline{CE}$), tendo a mesma funcionalidade.

A escolha da ativação por nível 0 deve-se, também, ao fato desta proporcionar maior **imunidade ao ruído**, pois, em situação contrária, haveria maior susceptibilidade para o acionamento dos blocos dentro do sistema, frente a este fator transiente indesejável.

8.6.3.1 Arquitetura Interna das Memórias ROM

Uma memória ROM, pode ser construída de inúmeras maneiras, porém, vamos estudar a arquitetura básica utilizada, principalmente, nos processos de fabricação dos circuitos integrados atuais. A figura 8.57 apresenta em blocos, a arquitetura básica de uma ROM genérica, com os respectivos terminais e barramentos de entrada e saída.

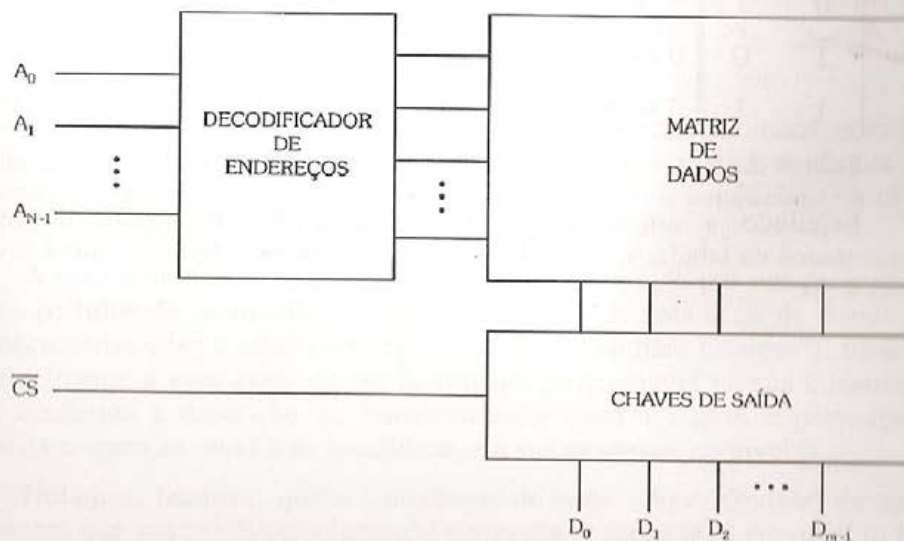



Figura 8.57

O primeiro bloco consiste num **decodificador de endereços**, que nada mais é que um gerador de produtos canônicos, responsável por ativar (fornecer nível 1) um fio de saída por vez, em função do endereçamento.

O segundo bloco é constituído por uma **matriz de dados**, que é um arranjo de linhas e colunas que, através de um elo de ligação, possibilita a

gravação dos dados pelo fabricante e conseqüente leitura pelo usuário. Na prática, dentre as várias tecnologias de construção, utilizam-se para a formação desses elos, elementos semicondutores (diodos ou transistores), que conforme visto a seguir, irão se constituir na estrutura de dados propriamente ditos.

Para a saída dos dados, a memória possui um conjunto de chaves (**Buffers**), que conforme habilitação através do terminal $\overline{CS}$, possibilita a conexão das saídas (nível 0), ou as deixa em alta impedância (nível 1), desconectando-as da barra de dados do sistema.

Para exemplificar, mostrando a estrutura de componentes interna aos blocos e explicar seu funcionamento, vamos construir uma ROM 4x8, com o conteúdo de dados presentes na tabela 8.15.

Endereço		**Dados**								
A_1	A_0	**Hex**	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	1 E	0	0	0	1	1	1	1	0
0	1	8 A	1	0	0	0	1	0	1	0
1	0	0 D	0	0	0	0	1	1	0	1
1	1	7 6	0	1	1	1	0	1	1	0

Tabela 8.15

Seguindo a arquitetura básica já vista e em função dos dados apresentados na tabela, vamos esquematizar a memória. Seu circuito é visto na figura 8.58.

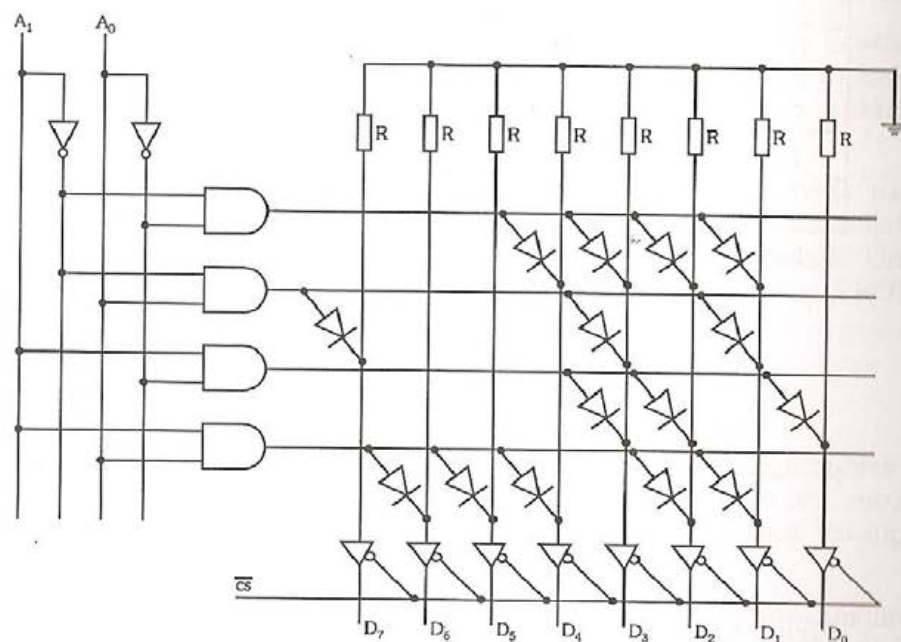



Figura 8.58

Pela figura, em confronto com a arquitetura básica em blocos, podemos facilmente identificar o decodificador de endereços, a matriz de dados e o conjunto de chaves de saídas (buffers), que ativos em 0, conduzirão os níveis relativos aos dados às saídas efetivas do bloco.

A matriz de dados, como já dissemos , é constituída por um conjunto de diodos (**células de memória**), formando na realidade uma série de portas OU confeccionadas com diodos (ver capítulo sobre "Famílias Lógicas"), trazendo principalmente a vantagem de ser facilmente programável na sua construção, pois, conforme a descrição do funcionamento visto a seguir, a presença do diodo dá origem ao nível 1 na localidade, e a sua ausência, ao nível 0.

Notamos, também, que a simbologia de cada chave (buffer) de saída, representa que sua habilitação (atuação como chave fechada) é em nível 0. Para melhor esclarecimento, a figura 8.59 mostra este elemento (a), e seu circuito equivalente (b), sendo sua atuação transcrita para a tabela 8.16.

M	Ch	E	S
0	FECHADA	0	0
		1	1
1	ABERTA	X	ALTA IMPEDÂNCIA

Tabela 8.16

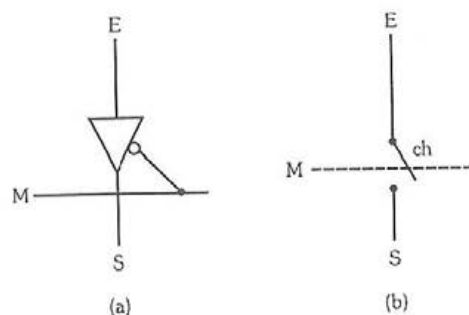



Figura 8.59

Para explicar o funcionamento da memória e mostrar como é obtido um dado de saída, vamos endereçar o caso 00 ($A_1 = 0$ e $A_0 = 0$), e aplicar um nível 0 à entrada $\overline{CS}$, para ativar todo o conjunto de chaves. Nesta situação de endereçamento, apenas o fio de saída da primeira porta é ativado (nível 1) pelo decodificador de endereços, provocando a condução de corrente dos respectivos diodos através dos resistores ao terra do circuito. Devido a estas conduções, surgirão nos resistores quedas de tensões que transpostas pelos fios até as saídas D_4, D_3, D_2, e D_1, resultarão no dado $1E_{16}$ (00011110_2). Convém observar que as tensões aparecem apenas nos fios com diodos colocados, sendo iguais a 0 nos outros, sem diodos.

Como outro exemplo, vamos endereçar o caso 10 ($A_1 = 1$ e $A_0 = 0$). Nesta situação, apenas o fio de saída da terceira porta é ativado (nível 1) pelo decodificador de endereços, surgindo quedas de tensões nos resistores pelos diodos que transpostas pelos fios até as saídas D_3, D_2 e D_0, resultarão no dado $0D_{16}$ (00001101_2).

Conforme a tecnologia de fabricação, são utilizados na matriz de dados ao invés de simples diodos, outros elementos semicondutores, tais como transistores bipolares ou transistores de efeito de campo (MOS-FET). Para

facilitar o processo de programação pelo fabricante, este utiliza um gabarito fotográfico das ligações elétricas chamado **máscara**, sendo as memórias assim confeccionadas denominadas **ROM Programadas por Máscara (Mask-Programmed Read-Only Memory)**.

As memórias ROM são produzidas com programações fixas para aplicações determinadas e sob encomenda, apenas em grande quantidade, normalmente para clientes específicos e fabricantes de equipamentos. Uma solução para o pequeno usuário é a utilização das ROMs programáveis (PROM e EPROM), estudadas nos itens a seguir.

8.6.4 Memórias PROM

As memórias PROM (**Programmable Read-Only Memory**) permitem o armazenamento dos dados pelo próprio usuário, porém feito de modo definitivo. Após esta programação, a memória PROM transforma-se em uma ROM, devendo, portanto, ser utilizada como tal.

O princípio básico da programação ou armazenamento de dados em uma PROM, é o de destruir, através de nível de tensão conveniente especificado pelo fabricante, as pequenas ligações semicondutoras existentes internamente nas localidades onde se quer armazenar a palavra de dados, conforme endereçamento feito. O roteiro para tanto é fornecido pelo fabricante nos manuais, sendo que, na prática, existem disponíveis sistemas apropriados (kits ou placas), para realizá-lo conforme o tipo de pastilha, com maior eficiência e rapidez. Devemos realçar que após a programação, o processo é irreversível, não sendo possível nenhuma alteração.

Este tipo de memória recebe, da mesma forma que a ROM, a classificação de não volátil, acesso aleatório e de apenas de leitura, pois apesar da programação prévia para a funcionalidade do sistema onde vai ser utilizada, só irá permitir a leitura de dados.

8.6.5 Memórias EPROM

Com o avanço da tecnologia, foram criadas as memórias EPROM (**Erasable Programmable Read-Only Memory**), ROM programável e apagável, que permitem a programação de modo semelhante à das PROMs, com a vantagem de poderem ser normalmente apagadas, mediante banho de luz ultravioleta, efetuado através da exposição da pastilha por uma janela existente em seu encapsulamento e, ainda, serem reprogramadas. São também conhecidas

como **UVPROM** (Ultraviolet PROM). Da mesma forma, após a programação, esta memória transforma-se em uma ROM, recebendo os mesmos itens de classificação.

As EPROMs são largamente utilizadas em circuitos digitais com microprocessadores, principalmente para o armazenamento de sistemas operacionais básicos residentes, responsáveis pelo funcionamento essencial do sistema, sobretudo no que se refere à conectividade elementar e funcional entre os circuitos integrados.

Convém ressaltar que o apagamento dos dados se dá de maneira simultânea e compacta para o programa inteiro, sendo necessária a regravação total do programa em caso de modificações por mais simples que sejam.

Existem disponíveis, comercialmente, vários tipos de EPROMs com diversas capacidades de armazenamento. Para exemplificar, mostrar a terminologia e a função dos terminais dos barramentos, sobretudo os básicos de controle, a figura 8.60 apresenta o bloco de uma EPROM do tipo mais comum, estruturada em 2Kx8.

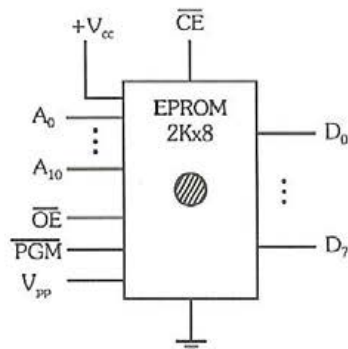



Figura 8.60

Identificação dos terminais:

⇨ A_0 - A_{10}: barra de endereços

⇨ D_0 - D_7: barra de dados

⇨ $\overline{CE}$: habilitação da pastilha (Chip Enable)

⇨ $\overline{OE}$: habilitação da saída (Output Enable)

⇨ $\overline{PGM}$: habilitação da programação (Program)

⇨ Vpp: tensão de programação (Program Supply Voltage)

Conforme a capacidade desta memória (2Kx8), para o acesso das localidades é necessário 11 fios (2^{11} = 2048=2K), e 8 para a barra de dados.

O terminal de habilitação $\overline{CE}$ tem a função de ativar o bloco através de nível 0, e quando em nível 1 o deixa desativado, na situação de baixo consumo de potência (**standby**). A entrada de controle $\overline{OE}$, por sua vez, tem a função de habilitar ou desabilitar apenas o barramento de saída, sendo da mesma forma a habilitação em nível 0.

Para a programação dos dados, o bloco dispõe de um terminal (Vpp), que recebendo uma tensão específica, sendo o valor fornecido pelo fabricante, é responsável juntamente com o terminal $\overline{PGM}$, pelo armazenamento das informações. O processo se realiza mediante a aplicação da tensão em Vpp (tipicamente um valor maior que Vcc), da habilitação da programação ($\overline{PGM}$) através de nível 0, do endereçamento e da aplicação das respectivas palavras de dados ao bloco, seqüencialmente, conforme a listagem do programa a ser armazenado.

O apagamento do programa pode ser feito pela exposição do bloco à luz ultravioleta durante 15 a 50 minutos, também conforme especificação dada pelo fabricante, em função da potência da lâmpada utilizada. Após o apagamento, todas as localidades assumem níveis 1, podendo o processo de regravação e apagamento se repetir por inúmeras vezes.

Na prática, da mesma forma, existem disponíveis sistemas apropriados (kits ou placas), para realizar o processo de gravação conforme o tipo de EPROM, com maior eficiência e rapidez. Existem, também, sistemas denominados apagadores de EPROM constituindo-se em uma caixa vedada, com lâmpada ultravioleta e sistema de cronometragem programada, que conforme o fabricante e especificação da EPROM, efetuam o processo de apagamento automaticamente.

8.6.6 Memórias EEPROM

As memórias **EEPROM** ou **E²PROM** (**Electrically Erasable Programmable Read-Only Memory**), constituem-se num avanço tecnológico em relação às EPROMs estudadas, pois permitem que o apagamento dos dados seja feito eletricamente e , ainda, isoladamente por palavra de dados, sem necessidade de reprogramação total. Este fato faz com que as alterações de programação sejam efetuadas pelo próprio sistema no qual a memória esteja inserida, sem necessidade de desconexão do circuito integrado, como no caso da EPROM.

Para ilustrar esta apresentação, a figura 8.61 apresenta o bloco de uma E^2 PROM de tipo comum, estruturada em 8Kx8.

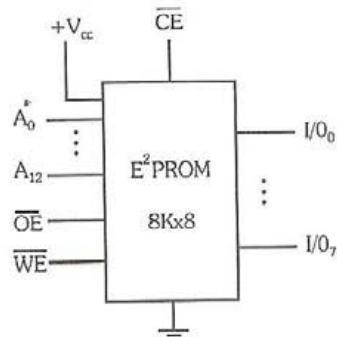



Figura 8.61

Notamos pela figura, que para o acesso das localidades desta memória é necessário 13 fios (2^{13} = 8192=8K). Notamos ainda, que devido à possibilidade de escrita e leitura pelos mesmos terminais, a barra de dados passa a ter a característica de bidirecional, recebendo a terminologia de I/O (Input/Output), muito comum nestes casos.

A escrita de uma palavra de dados, alterando a programação, é obtida através do endereçamento e respectiva aplicação da palavra nos terminais da barra de dados, isto com o terminal $\overline{OE}$ em nível 1, e o de habilitação da escrita $\overline{WE}$ (**Write Enable**), em nível 0, dentro de um ciclo de tempo mínimo, especificado em manual pelo fabricante do circuito integrado.

Em nível de classificação, esta memória pode causar polêmica em um item, pois apesar de permitir a escrita e leitura de dados, faz parte da família das memórias apenas de leitura (ROM). O nome EEPROM, no entanto, deve ter sido atribuído por questões históricas dentro do desenvolvimento tecnológico na área, o mesmo ocorrendo com outras memórias. Além deste item, esta memória é não volátil e possui acesso aleatório.

8.6.7 Memórias RAM

As memórias RAM, conforme já mencionado, permitem a escrita e leitura dos dados e possuem acesso aleatório ou randômico. Vem daí o nome RAM (Random-Access Memory). Além disso, são voláteis, pois perdem seus dados armazenados com o desligamento da alimentação. Possuem, ainda, um tempo de acesso muito reduzido, sendo utilizadas em equipamentos digitais principalmente como memórias de programas e dados para armazenamento de

forma temporária, pois , em função de volatilidade, estes são perdidos no desligamento ou interrupção da energia.

Quanto ao armazenamento, são encontradas nos tipos estáticas (**SRAM: Static RAM**), ou dinâmicas (**DRAM: Dynamic RAM**). As RAMs estáticas utilizam como célula básica de memória o flip-flop, possuindo em sua arquitetura vários elementos. Já as do tipo dinâmicas possuem circuitos mais simples, porém necessitando de reinserção de dados periódica em ciclo, na prática denominada **refresh** (termo em inglês, que significa "refrescar"), sendo esta operação controlada pelo microprocessador do sistema. A célula básica da RAM dinâmica armazena cada dado por efeito capacitivo do pequeno semicondutor formado internamente, por este motivo, apresenta a vantagem de alta capacidade de armazenamento por circuito integrado.

A figura 8.62 apresenta o bloco representativo de uma memória RAM estática, com terminais e barramentos já conhecidos e mais um terminal de controle **$R/\overline{W}$ (Read/Write)** de dupla função, para possibilitar a leitura ($R/\overline{W}=1$), ou escrita ($R/\overline{W}=0$) dos dados nas localidades endereçadas.

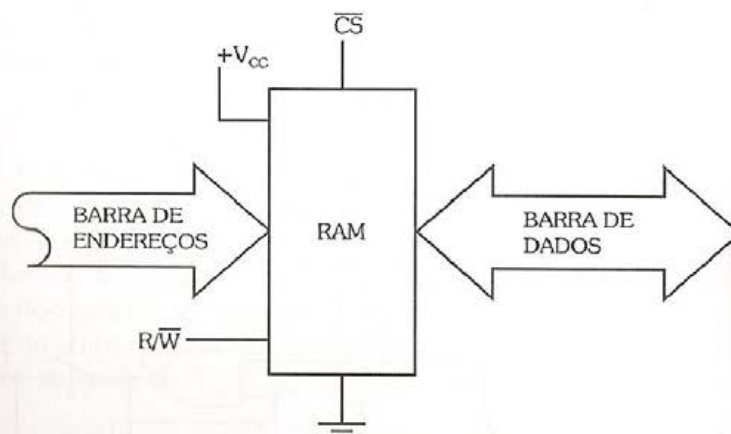



Figura 8.62

Para entendermos o funcionamento básico de uma RAM estática, vamos, inicialmente analisar o circuito de uma célula básica que permite a escrita e leitura de 1 bit de informação. Este circuito é visto na figura 8.63.

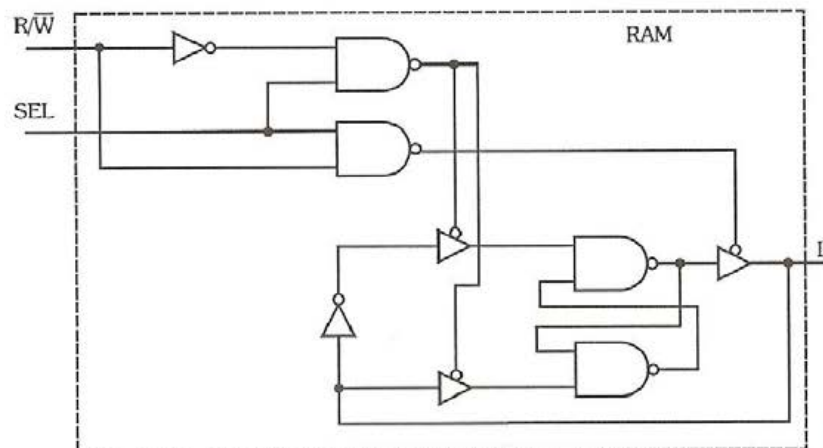



Figura 8.63

Para efetuar a escrita de um dado, devemos primeiramente selecionar a célula (SEL=1) e passar o controle de leitura/escrita (R/$\overline{\text{W}}$) para 0. Logo após, aplicamos o dado no terminal D, agora configurado como entrada.

A figura 8.64 apresenta a célula básica, com todas estas situações colocadas e, ainda como exemplo, a aplicação para armazenamento de nível 1 na entrada D.

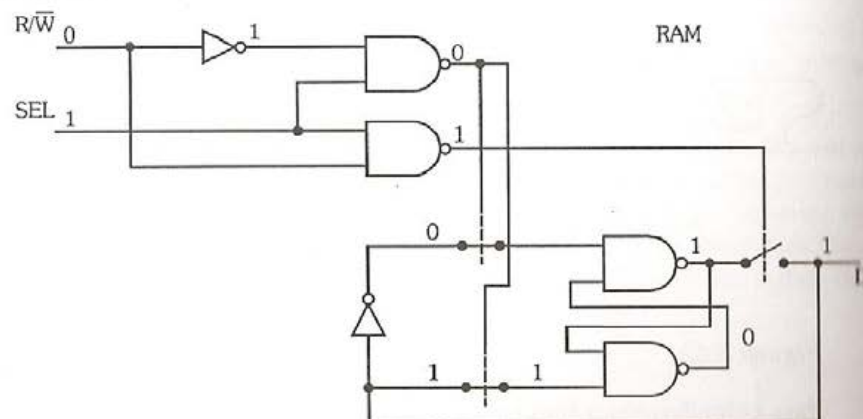
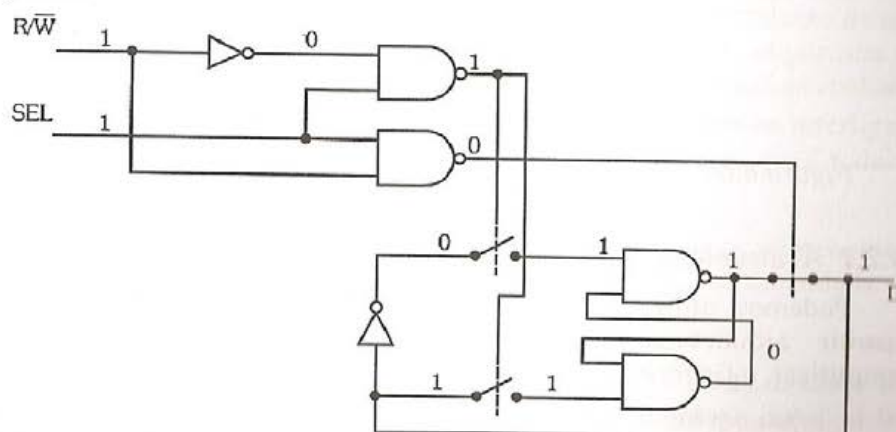



Figura 8.64

Pela figura notamos que a porta NE superior irá, através de nível 0 em sua saída, ativar as duas chaves (buffers), aqui substituídas pelos circuitos equivalentes, fazendo o dado ser aplicado ao flip-flop e conseqüentemente ser

armazenado na saída. Enquanto isso, a outra porta, através de nível 1 em sua saída, irá desativar a chave de saída, permitindo a escrita ou entrada do dado.

Para efetuar a leitura, devemos também selecionar a célula (SEL=1), e passar o controle $R/\overline{W}$ para 1, sendo obtido o dado armazenado pelo terminal D, agora configurado como saída. A figura 8.65 mostra esta situação colocada no esquema.

Figura 8.65

Pela figura, notamos que a porta NE superior, através de nível 1 em sua saída, irá desativar as chaves nas entradas das portas do flip-flop, impedindo a escrita de um novo dado. Estas entradas, estando em vazio devido à abertura das chaves, assumem nível 1 (ver capítulo sobre "Famílias Lógicas"), fazendo o flip-flop manter as suas saídas Qf = Qa (circuito elementar de um flip-flop RS). Enquanto isso, a outra porta NE através de nível 0 em sua saída, irá ativar a chave na saída do flip-flop, fazendo o dado armazenado ser transposto à saída D.

No caso da célula não ser selecionada (SEL = 0), as 2 portas NE apresentarão nível 1 em suas saídas, mantendo as três chaves abertas, deixando a célula com a saída desativada (tri-state) , impedindo qualquer escrita ou leitura de dados.

Na realidade, dentro dos circuitos integrados, são construídas células básicas com diversas configurações tecnológicas e de circuitos, sendo esta apresentada devido ao seu enorme caráter didático. Nos itens seguintes, para facilitar, utilizaremos para desenvolver a arquitetura interna deste tipo de memória , células genéricas representadas em blocos. A figura 8.66 mostra o

bloco padrão representativo da célula da memória RAM, sendo sua atuação resumida na tabela 8.17.

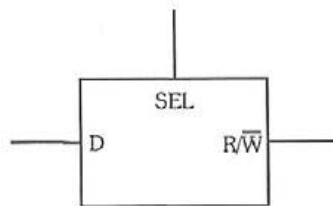



Figura 8.66

SEL	R/W̄	D
0	X	TRI-STATE
1	0	ENTRADA PARA ESCRITA
	1	SAÍDA PARA LEITURA

Tabela 8.17

8.6.7.1 Arquitetura Interna das Memórias RAM

Podemos, utilizando a célula básica padrão analisada no item anterior, construir arquiteturas de memórias RAM estáticas no formato Nxm. Para exemplificar, a figura 8.67 mostra a arquitetura de uma RAM de estrutura 4x4.

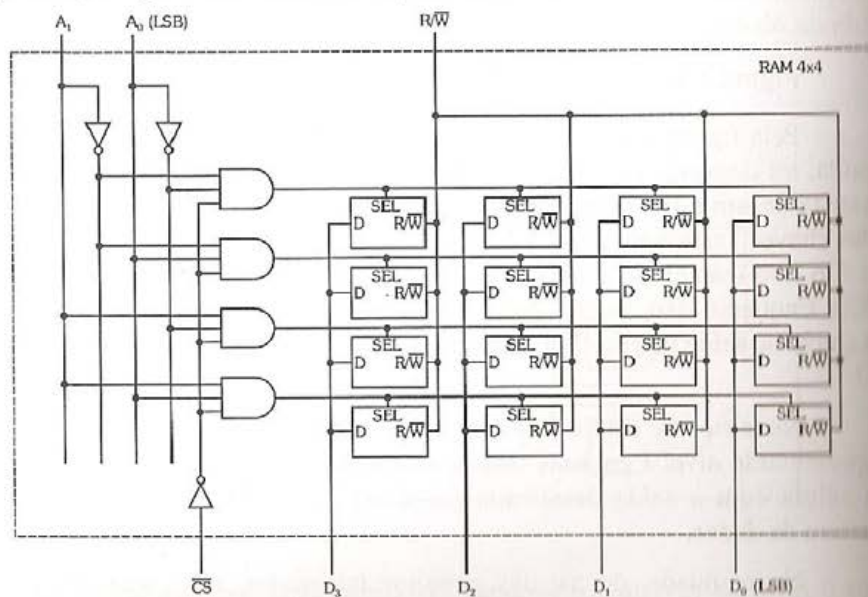



Figura 8.67

Uma RAM, assim especificada, possui quatro localidades com 4 bits cada.

O circuito, como se observa, é constituído por um decodificador de endereços com dois fios (A_1 e A_0), responsável pelo endereçamento de cada localidade definida pelo conjunto das quatro células interligadas horizontalmente. Os terminais de dados (D) estão também interligados, porém, por posicionamento do bit na palavra de dados, pois no endereçamento de cada conjunto através das entradas SEL, os outros não endereçados adquirirem nos terminais D, a situação de alta impedância (tri-state), sendo desconectados do fio em comum. Além disso, todas as entradas $R/\overline{W}$ encontram-se interligadas para propiciar um controle simultâneo da escrita ou leitura para todas as localidades.

Para mostrar o funcionamento desta memória, vamos primeiramente efetuar o armazenamento (escrita) do dado 5_{16} (0101_2), na localidade 1_{16}, endereçada por 01.

Inicialmente, estando a pastilha não selecionada ($\overline{CS}=1$), o nível 0 na entrada das portas E após o inversor, ocasiona o aparecimento de nível 0 na saída destas, fazendo todas as células de memória entrar em estado de alta impedância (SEL=0 $\Rightarrow$ D em tri-state).

Feita a seleção da pastilha ($\overline{CS}=0$) e o endereçamento da localidade ($A_1=0$ e $A_0=1$), o segundo fio superior na saída da porta E irá, mediante nível 1, selecionar todas as células da linha (SEL=1). Com o controle $R/\overline{W}$ em 0 (escrita), aplicamos os dados nos respectivos terminais, agora configurados como entradas ($D_3=0$, $D_2=1$, $D_1=0$ e $D_0=1$), sendo estes armazenados pelas células.

Com a passagem de $R/\overline{W}$ para 1, para posterior leitura, os dados irão permanecer armazenados, mesmo na reversão da seleção da célula com $\overline{CS}$ passando para nível 1. Devemos ressaltar ainda, que a informação será perdida caso se desligue a tensão de alimentação.

O mesmo processo de escrita pode ser estendido para outras localidades, bastando endereçar, passando $R/\overline{W}$ para 0 e aplicando respectivamente a informação de dados nas entradas D.

Para a leitura de uma informação, devemos selecionar a pastilha ($\overline{CS}=0$), e com $R/\overline{W}$ igual a 1, endereçar a localidade, obtendo assim a informação de dados nos terminais D, agora configurados como saídas.

A figura 8.68 mostra a representação em bloco da RAM utilizada no exemplo.

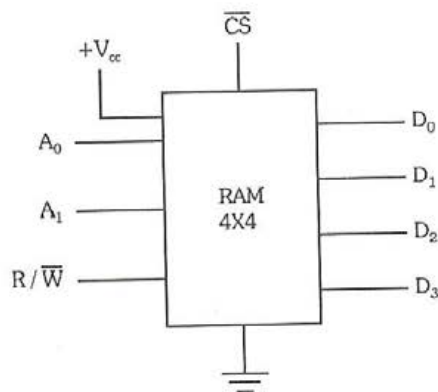



Figura 8.68

8.6.7.2 Expansão da Capacidade da Memória RAM

Em certas aplicações, é comum necessitarmos de memórias de maior capacidade ou diferentes das encontradas no mercado. Neste tópico, mostraremos como expandir a capacidade de uma RAM, caso muito comum na prática, porém valendo o mesmo processo também para outras memórias.

A expansão pode ser obtida pela palavra de dados, pelo aumento de localidades ou ainda por ambos, conforme a situação.

Para iniciarmos esta análise, vamos elaborar um exemplo simples que mostra a expansão da palavra de dados. Vamos formar uma memória RAM 256x8, a partir de dois blocos de estrutura 256x4. A figura 8.69 mostra a ligação dos blocos que possibilita esta expansão.

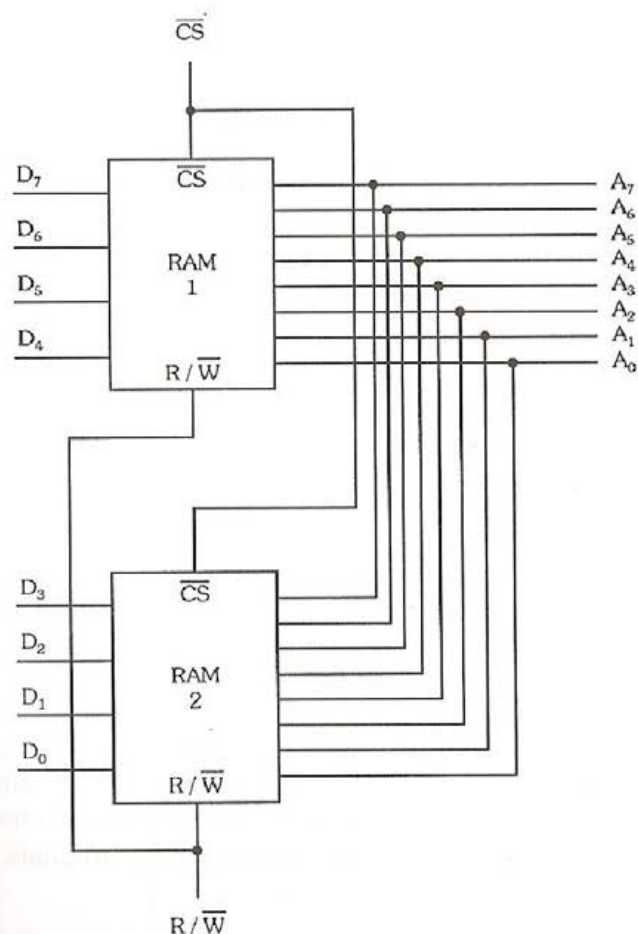



Figura 8.69

Notamos pela figura, que os terminais de endereçamento (A_7 a A_0), de seleção da pastilha ($\overline{CS}$) e de controle de escrita/leitura (R / $\overline{W}$) são interligados, pois estas operações são comuns aos dois blocos na memória obtida. A barra de dados, porém, é composta pela associação da barra de cada memória (4 Bits), resultando em uma palavra de dados maior (8 bits de D_7 a D_0), aumentando assim a capacidade da memória .

Nesta memória derivada, o endereço da localidade inicial é 00_{16} (00000000_2), e o da final FF_{16} (11111111_2).

Para exemplificar sua atuação, vamos efetuar o armazenamento do dado 28_{16} (00101000_2) na localidade endereçada por $3E_{16}$ (00111110_2).

Primeiramente, selecionamos a pastilha ($\overline{CS}$=0), logo após, aplicamos os níveis 00111110 ($3E_{16}$) à barra de endereços e, a seguir, passamos o controle de escrita/leitura ($R / \overline{W}$) para 0, e por último, aplicamos os níveis 00101000 (28_{16}) à barra de dados.

É fácil perceber que o armazenamento da informação na memória expandida será feito em duas partes, sendo a mais significativa (0010) na RAM 1, e a menos significativa (1000) na RAM 2, isso com endereçamento simultâneo às duas localidades.

Para a leitura da informação, devemos passar $R / \overline{W}$ para 1 e endereçar a localidade, obtendo assim a informação de dados nos terminais de D_7 a D_0, simultaneamente, agora configurados como saídas.

Como já dissemos, uma outra forma possível de expansão da capacidade é com aumento das localidades de memórias. Para exemplificar este processo, vamos formar uma memória RAM 128x4, utilizando blocos de estrutura 32x4.

Para funcionar na forma 128x4, o sistema deve ser composto por 4 RAMs de 32x4, sendo o endereçamento feito por 7 terminais (2^7 = 128). Para tanto, utilizaremos os 5 terminais de cada bloco interligados e como complemento, associaremos 2 fios auxiliares, que através de um circuito apropriado, farão a seleção de cada bloco em seqüência, através das entradas $\overline{CS}$. Na parte relativa aos dados, basta interligar os respectivos terminais, pois não há expansão.

A figura 8.70 mostra a ligação dos quatro blocos e o circuito seletivo que possibilita esta expansão.

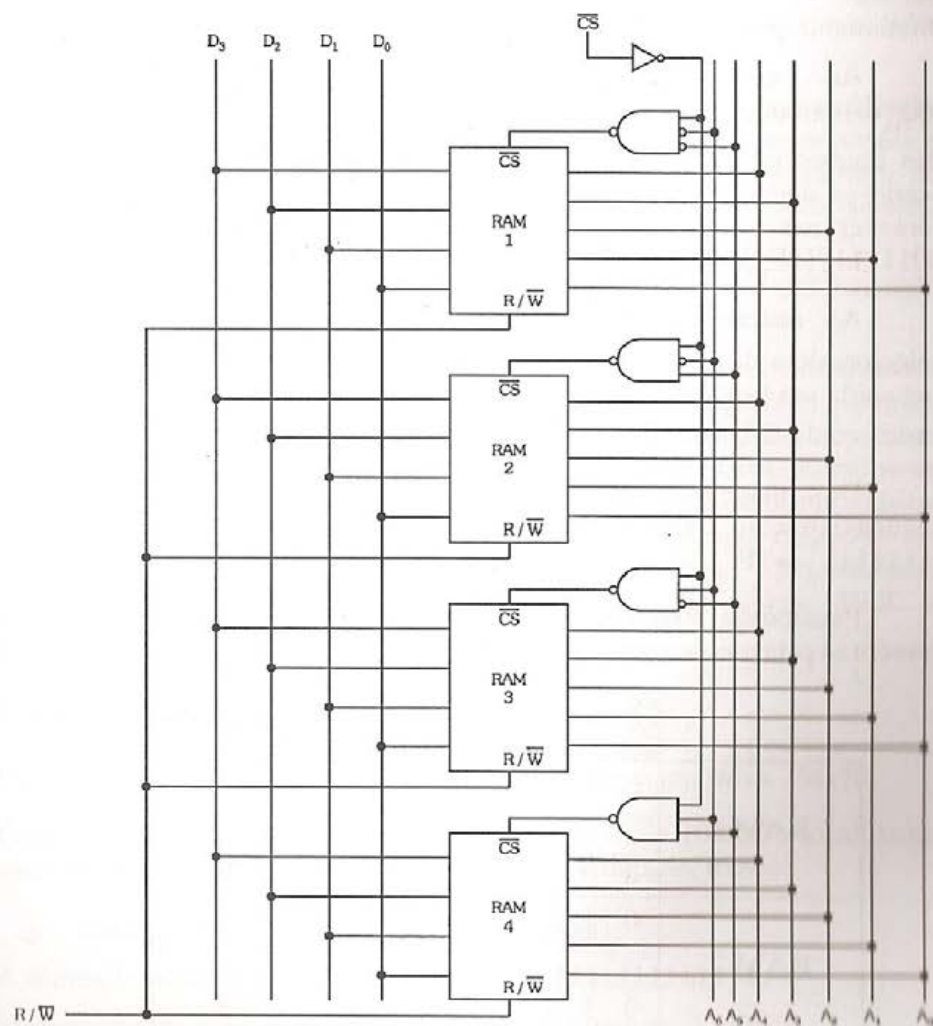



Figura 8.70

Estando o sistema não selecionado ($\overline{CS}$ =1), o nível 0 na entrada das portas NE após o inversor, ocasiona o aparecimento de níveis 1 nas saídas destas, fazendo todos os blocos de memória entrar em estado de alta impedância (terminais de dados em tri-state).

Feita a seleção ($\overline{CS}$ =0), o nível 1 na entrada das portas irá liberá-las para transmitir o endereçamento aos blocos. A palavra de endereço das localidades será, no sistema expandido, composta pelos cinco fios de cada bloco

interligados (A_0 a A_4), e mais dois fios complementares responsáveis diretamente pelos níveis de saída das portas, para efetuar a seleção parcial.

Ao efetuarmos o endereçamento da primeira localidade ($00_{16} \Rightarrow 0000000_2$), a porta NE relativa à RAM 1, em função de A_6=0 e A_5=0 e dos inversores ligados, irá selecionar a RAM 1 através de nível 0 em $\overline{CS}$, sendo as demais memórias não selecionadas, pois as respectivas portas irão fornecer nas saídas níveis 1. Com isso, seqüencialmente, até o endereço 0011111_2 ($1F_{16}$), estaremos ocupando as localidades relativas à RAM 1.

Ao endereçarmos a localidade seguinte ($0100000_2 \Rightarrow 20_{16}$), teremos selecionado a da RAM 2, pois a respectiva porta NE irá apresentar nível 0 em sua saída em função de A_6=0 e A_5=1, estando as demais RAMs desativadas. O endereço da RAM 2 vai até 0111111_2 ($3F_{16}$).

De maneira análoga, o processo de endereçamento segue seqüencialmente, bloco a bloco, até a última localidade da memória expandida ($1111111_2 \Rightarrow 7F_{16}$).

Para mostrar toda a estrutura de endereçamento desta memória, a tabela 8.18 contém as palavras de endereço inicial e final de cada RAM integrada ao sistema.

	A_6 A_5 A_4 A_3 A_2 A_1 A_0	Hex
RAM 1	0 0 0 0 0 0 0	00
	0 0 1 1 1 1 1	1F
RAM 2	0 1 0 0 0 0 0	20
	0 1 1 1 1 1 1	3F
RAM 3	1 0 0 0 0 0 0	40
	1 0 1 1 1 1 1	5F
RAM 4	1 1 0 0 0 0 0	60
	1 1 1 1 1 1 1	7F

Tabela 8.18

Para se efetuar a escrita ou a leitura de dados neste sistema expandido, procede-se normalmente, conforme já explicado.

Uma outra possibilidade de expansão de memória consiste na ampliação de palavra de dados e também no número de localidades. Este processo é obtido através da fusão dos dois aqui abordados aumenta-se o número de localidades utilizando o circuito seletivo e associa-se um outro sistema semelhante para compor a nova palavra de dados.

No item relativo aos exercícios resolvidos, mostraremos um exemplo desse procedimento.

Na área dos microcomputadores, devido principalmente à flexibilidade de montagem das configurações de hardware e à rápida evolução dos sistemas, os equipamentos possuem na placa principal uma série de conectores livres (**slots**), sendo alguns destinados exclusivamente a receber novos módulos de memórias RAM (DRAM), para serem utilizados como expansão. Na prática, estes módulos constituem-se em arranjos destas memórias, que dispostas em pequenas plaquetas de circuito impresso removíveis denominadas popularmente **pentes de memória**, possibilitam de forma fácil a expansibilidade do sistema.

8.6.8 Exercícios Resolvidos

1- Determine a palavra de endereço inicial e final de uma memória de 1Mx16.

Com esta especificação, a memória irá possuir 2^{20} = 1048576 localidades com 16 bits, endereçadas de 0 a 1048575_{10}. Em binário, temos:

⇨ 1048576 =2^{20} = 100000000000000000000

A última localidade é: 1048575 = 011111111111111111111

Transformando diretamente para hexadecimal, temos: $FFFFF_{16}$.

∴ A palavra de endereço inicial é 00000_{16}, e a final $FFFFF_{16}$.

2- Determine o mapeamento de uma memória ROM para atuar com decodificador do código BCD 8421 para o 2 entre 5. Calcule a capacidade de memória.

Nesta situação, a ROM irá funcionar como um circuito combinacional (decodificador), sendo o código de entrada injetado nas variáveis de endereçamento e o de saída fixado nas correspondentes localidades de memória. A figura 8.71 mostra a representação do bloco como decodificador.

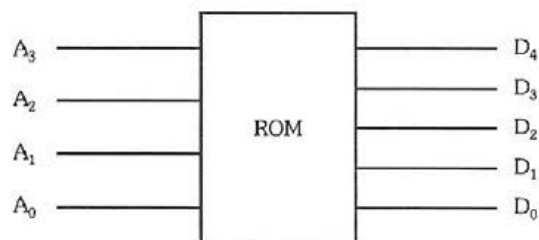



Figura 8.71

A partir destas informações, vamos montar a tabela com o mapeamento parcial da memória:

ENDEREÇAMENTO				DADOS				
BCD 8421				2 ENTRE 5				
A_3	A_2	A_1	A_0	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	1	1
0	0	0	1	0	0	1	0	1
0	0	1	0	0	0	1	1	0
0	0	1	1	0	1	0	0	1
0	1	0	0	0	1	0	1	0
0	1	0	1	0	1	1	0	0
0	1	1	0	1	0	0	0	1
0	1	1	1	1	0	0	1	0
1	0	0	0	1	0	1	0	0
1	0	0	1	1	1	0	0	0

Tabela 8.19

Podemos notar pela tabela que, para executar esta função a memória necessita dez localidades com 5 bits cada. Sendo a memória total especificada por 16x5, pois quatro variáveis possibilitam dezesseis localidades, sua capacidade será, então, de 80 bits.

As localidades não endereçadas, por não pertencerem ao código, podem ser preenchidas com todos os dados iguais a 1.

3- Utilizando blocos de memória RAM 128x4, forme uma de 256x8. Escreva a palavra de endereço inicial e final de cada bloco integrado ao sistema.

Esta expansão consiste na ampliação tanto do número de localidades, como também, da palavra de dados. Para executá-la, vamos determinar inicialmente o número de terminais da nova memória. Assim sendo, temos:

Localidades: $256 = 2^8 \Rightarrow$ oito terminais de endereçamento.

Dados: oito terminais de dados.

Para o endereçamento, vamos utilizar os sete terminais de cada bloco ($128 = 2^7$), e mais um fio ligado a um circuito seletivo. Para compor a nova palavra de dados, associa-se um outro sistema semelhante para a duplicação. A figura 8.72 mostra a ligação dos blocos e o circuito seletivo que possibilita esta expansão.

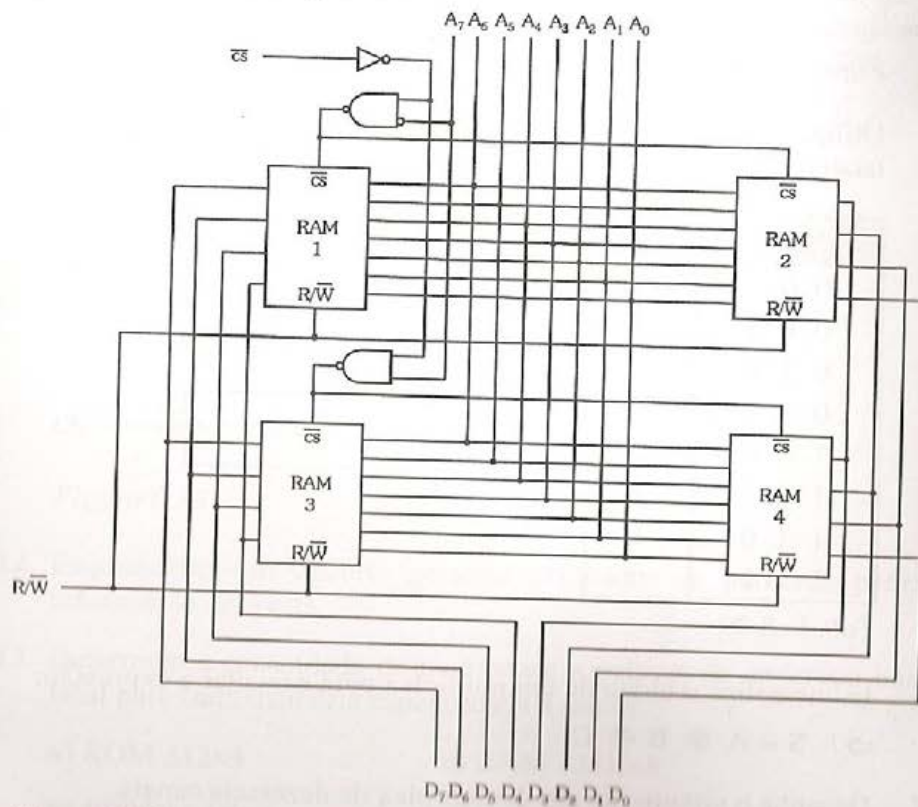



Figura 8.72

8.7 Exercícios Propostos

8.7.1 Calcule o número de portas E necessárias para construir um multiplex de dezesseis canais utilizando matriz de encadeamento simples?

8.7.2 Repita o exercício anterior, utilizando uma matriz de encadeamento duplo.

8.7.3 Utilizando cinco blocos multiplex de oito canais, esquematize um sistema multiplex de 32 canais.

8.7.4 A figura 8.73 apresenta os sinais de seleção e de informação de entrada de um multiplex de dois canais. Esboce o sinal multiplexado.

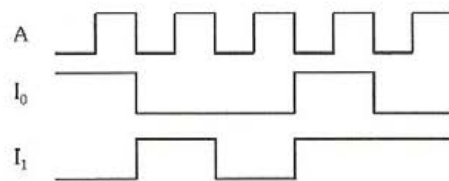



Figura 8.73

8.7.5 Utilizando o bloco de um multiplex, elabore o circuito que executa a tabela 8.20.

A B C	S
0 0 0	0
0 0 1	0
0 1 0	1
0 1 1	1
1 0 0	0
1 0 1	1
1 1 0	1
1 1 1	0

Tabela 8.20

8.7.6 Esquematize o bloco de um multiplex para executar a expressão:

⇨ $S = A \oplus B \oplus C$

8.7.7 Desenhe o circuito de um demultiplex de dezesseis canais.

8.7.8 A partir de dois blocos demultiplex de dezesseis canais e um de dois canais, forme um sistema demultiplex de trinta e dois canais.

8.7.9 A figura 8.74 apresenta os sinais de seleção e de entrada multiplexada de um demultiplex de dois canais. Esboce os sinais de informação.

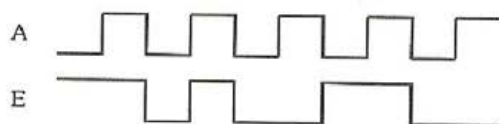



Figura 8.74

8.7.10 Utilizando o bloco de um demultiplex, elabore um decodificador 4 para 16, onde apenas uma saída é ativada para cada combinação de entrada.

8.7.11 Determine os gráficos de saída para o sistema esquematizado na figura 8.75, sabendo-se que o nível 1 corresponde a 5V e que a freqüência de clock é 2 MHz.

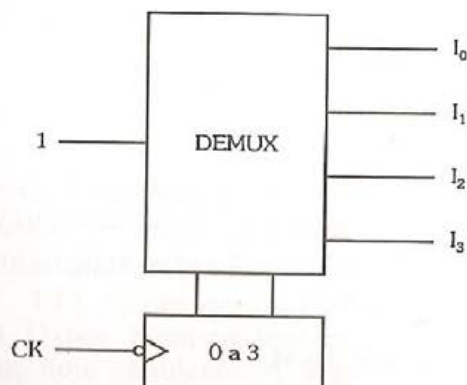



Figura 8.75

8.7.12 Esquematize um circuito gerador/verificador de paridade para uma informação de 3 bits.

8.7.13 Determine a capacidade de memória e a palavra de endereço inicial e final para cada memória especificada a seguir:

a) ROM 512x4 c) RAM 128Kx8

b) EPROM 4Kx8 d) RAM 2Mx16

8.7.14 Esquematize o circuito interno de uma ROM, com o seguinte conteúdo:

⇨ 01_{16}, $3F_{16}$, 23_{16}, $4B_{16}$, 56_{16}, 88_{16}, $9C_{16}$ e ED_{16}

8.7.15 Determine o mapeamento de uma memória PROM para atuar como gerador de caracteres para hexadecimal, ou seja, a partir de um código binário, forneça os níveis para fazer um display de sete segmentos catodo comum apresentar a seqüência do sistema hexadecimal. Especifique a memória e determine sua capacidade.

8.7.16 Determine o mapeamento de uma memória EPROM para, através do sistema gerador de funções esquematizado, gerar a função digitalizada vista na figura 8.76. Calcule a freqüência de clock do contador.

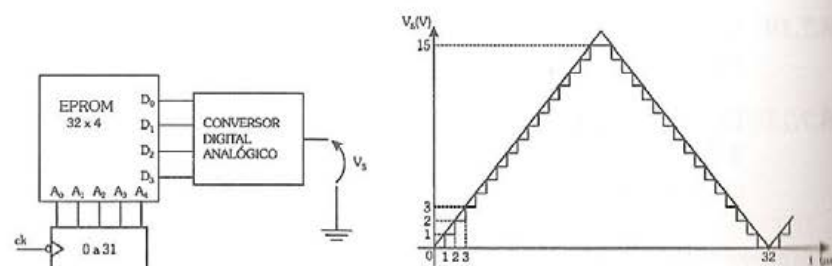



Figura 8.76

8.7.17 Utilizando apenas portas NOU e inversores, modifique o circuito da célula básica da memória RAM do item 8.6.7, para este executar as mesmas funções. Considere que nas portas utilizadas, cada terminal em vazio equivale a nível lógico 1.

8.7.18 Desenhe a arquitetura interna de uma RAM 8x1.

8.7.19 A partir de blocos RAM 64x4, esquematize uma RAM 64x8. Escreva as palavras de endereçamento inicial e final de cada RAM integrada ao sistema.

8.7.20 Idem ao exercício anterior, para obter uma RAM 512x4 com blocos de estrutura 128x4.

8.7.21 Idem, para uma RAM 64x8, obtida com blocos 32x4.

CAPÍTULO 9

Famílias de Circuitos Lógicos

9.1 Introdução

Até aqui, utilizamos os blocos lógicos sem nos preocuparmos com suas estruturas internas. Dedicaremos este capítulo ao estudo das principais famílias de circuitos lógicos utilizadas atualmente.

Entende-se por famílias de circuitos lógicos, os tipos de estruturas internas que nos permitem a confecção destes blocos em circuitos integrados. Cada família lógica utiliza determinados componentes em seus blocos e, de acordo com estes, a família possuirá determinadas características relacionadas ao seu funcionamento e desempenho prático.

As famílias utilizadas atualmente dentro da área de Eletrônica Digital são a **TTL (Transistor-Transistor Logic)** e a **CMOS (Complementary Metal Oxide Semiconductor)**, porém derivam de uma série de famílias lógicas, hoje obsoletas. A seguir, vamos relacionar, em escala tecnológica evolutiva, algumas famílias utilizadas anteriormente, precedentes à família TTL:

- ⇨ DCTL (Direct-Coupled Transistor Logic)
- ⇨ RTL (Resistor-Transistor Logic)
- ⇨ RCTL (Resistor-Capacitor Transistor Logic)
- ⇨ DTL (Diode-Transistor Logic)
- ⇨ HTL (High-Threshold Logic)
- ⇨ ECL (Emitter-Coupled Logic)

O estudo das características da maioria destas famílias citadas não faz sentido nos dias de hoje, a não ser que seja feito com aspectos de evolução histórica, mostrando a origem construtiva da tecnologia atual.

A família ECL, em particular, embora não tenha sido desenvolvida na atualidade, ainda é utilizada devido principalmente ao seu comportamento frente a situações que exigem alta velocidade de operação, característica típica desta família, sendo, porém, seu emprego restrito a aplicações específicas, não se caracterizando mais em série comercial.

Estudaremos, mais adiante, as famílias TTL e CMOS, e as respectivas versões derivadas. Primeiramente, vamos abordar alguns conceitos básicos para melhor compreensão e avaliação das mesmas.

9.2 Conceitos e Parâmetros das Famílias Lógicas

Vamos, nos itens subseqüentes, abordar os principais conceitos evolvidos no estudo das famílias de circuitos lógicos. São tópicos que caracterizam parâmetros como os níveis de tensão e de corrente de entrada e saída, quantidades de blocos a serem conectados, tempo de resposta do bloco e seu fator de imunidade ao ruído. O primeiro conceito a ser abordado é relativo aos níveis de tensão e de corrente.

9.2.1 Níveis de Tensão e de Corrente

No capítulo 2, definimos nível 1 e nível 0. Na realidade, esses níveis irão variar dentro de faixas. O nível 0 não precisa ser necessariamente 0, mas, sim, uma tensão pequena abaixo de um certo valor máximo. O nível 1, como foi definido, representa uma tensão, mas não precisa ser necessariamente um valor e, sim, uma faixa acima de um valor mínimo e abaixo de um valor máximo. Conforme a tecnologia de construção do circuito interno, cada família ou versão derivada irá possuir uma faixa de trabalho para esses níveis, sendo especificações diferentes para entrada e saída do bloco.

Um outro parâmetro é o que trata da corrente. Quando um nível lógico 1 for aplicado a uma entrada de um bloco lógico, esta irá consumir uma corrente. O mesmo ocorre quando a saída de um bloco lógico em nível 1 for conectada à entrada de outro. Haverá uma drenagem de corrente, na prática limitada.

Da mesma forma, se for aplicado o nível 0 (potencial de terra) à entrada de um bloco lógico, haverá uma deriva de corrente, no sentido do bloco para o terminal, originada conforme as características do circuito do bloco. A saída,

por sua vez, em nível 0, irá também absorver uma corrente originária da entrada do bloco seguinte conectado.

Existe uma terminologia padrão empregada pelos principais fabricantes de circuitos integrados nos respectivos manuais, para designar estes parâmetros. Vamos apresentá-los e defini-los, a seguir:

⇨ V_{IL} (**Low-level Input Voltage**): valor de tensão (máxima), que garante o nível 0 na entrada.

⇨ V_{OL} (**Low-level Output Voltage**): valor de tensão (máxima), que garante o nível 0 na saída.

⇨ V_{IH} (**High-level Input Voltage**): valor de tensão (mínima), que garante o nível 1 na entrada.

⇨ V_{OH} (**High-level Output Voltage**): valor de tensão (mínima), que garante o nível 1 na saída.

⇨ I_{IL} (**Low-level Input Current**): valor de corrente (máxima), no terminal de entrada (no sentido do bloco para o terminal), quando é aplicado o nível 0.

⇨ I_{OL} (**Low-level Output Current**): valor de corrente (máxima), que a saída pode receber quando em nível 0.

⇨ I_{IH} (**High-level Input Current**): valor de corrente de entrada (máxima), quando é aplicado nível 1.

⇨ I_{OH} (**High-level Output Current**): valor de corrente de saída (máxima), quando em nível 1.

Nos manuais, além dos limites de mínimo e máximo, conforme a definição do parâmetro, são encontrados os valores típicos de trabalho.

A figura 9.1 apresenta os diagramas relativos aos níveis de tensão definidos, tanto para a entrada (a), como para a saída (b), de um mesmo bloco lógico.

(a) (b)

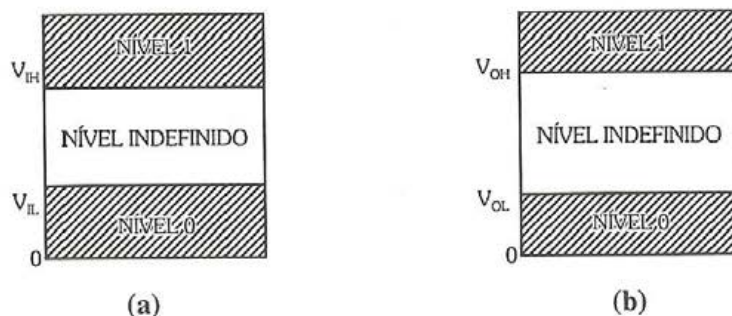



Figura 9.1

Notamos que na região compreendida entre o valor máximo de nível 0 (V_{IL} e V_{OL}), e o valor mínimo de nível 1 (V_{IH} e V_{OH}) o nível lógico será indefinido. Para existir compatibilidade com segurança entre entrada e saída na família, é necessário que V_{OL} seja menor que V_{IL} , e que V_{OH} seja maior que V_{IH}.

9.2.2 Fan-Out

Até agora, trabalhamos com os blocos lógicos sem nos preocuparmos com o número de conexões feitas nas saídas. Na prática, existe um parâmetro denominado, em inglês, de **Fan-Out** (feixe de saída), que estipula o limite dessas ligações.

Definimos Fan-Out como sendo o número máximo de blocos lógicos que pode ser ligado à saída de outro da mesma família. Embora esta definição seja em nível de mesma família lógica, este fator pode ser determinado entre famílias e versões compatíveis.

Se este fator for excedido na ligação da saída de um bloco às entradas de outros, os limites máximos de corrente serão ultrapassados, acarretando principalmente a queda do nível 1 de saída.

O Fan-Out está relacionado com as correntes máximas de saída e de entrada dos blocos lógicos, podendo ser determinado no nível 0 e no nível 1. A seguir, vamos escrever as relações para estes cálculos:

$$\text{Fan-Out}_{(\text{nível } 0)} = \frac{I_{OL}}{I_{IL}} \qquad \text{e} \qquad \text{Fan-Out}_{(\text{nível } 1)} = \frac{I_{OH}}{I_{IH}}$$

Os valores de corrente utilizados nas relações são extraídos dos manuais comerciais.

Na prática, os fabricantes de circuitos integrados, normalmente, generalizam o valor obtido para um só, válido para toda a família lógica, porém pode haver variação deste valor conforme a versão lógica utilizada.

Para exemplificar este procedimento, vamos calcular estes parâmetros para os blocos de um circuito integrado muito comum da família TTL. A tabela 9.1 apresenta as especificações do circuito integrado TTL 7400 (4 portas NE de 2 entradas na versão padrão).

TTL	*7400*	
Parâmetros	**Valores Máximos**	**Unidade**
I_{OL}	16	mA
I_{IL}	1,6	mA
I_{OH}	400	μA
I_{IH}	40	μA

Tabela 9.1

Utilizando as relações para os valores da tabela, temos:

$$\text{Fan-Out}_{(\text{nível } 0)} = \frac{I_{OL}}{I_{IL}} = \frac{16}{1,6} = 10$$

$$\text{Fan-Out}_{(\text{nível } 1)} = \frac{I_{OH}}{I_{IH}} = \frac{400}{40} = 10$$

Concluímos pelos resultados que à saída deste bloco, poderemos ligar no máximo outros 10, ou seja, 10 terminais de entrada de blocos similares.

9.2.3 Tempo de Atraso de Propagação

O **tempo de atraso de propagação (propagation delay time)** é definido como o tempo que um bloco lógico leva para mudar de estado desde a aplicação de um nível lógico para tanto. Em outras palavras, é o tempo que um bloco leva para responder, ou seja, passar do estado 1 para o estado 0 ou vice-versa. Na prática, em terminologia de manuais, o tempo de atraso de propagação quando vai do nível 0

para 1, é representado por t_{PLH} (Low to High), e quando vai de 1 para 0 por t_{PHL} (High to Low), e seu valor é da ordem de nanossegundos (ns).

Para ilustrar, a figura 9.2 apresenta um inversor com exemplos de trechos de sinais aplicados à entrada e os respectivos resultados de saída.

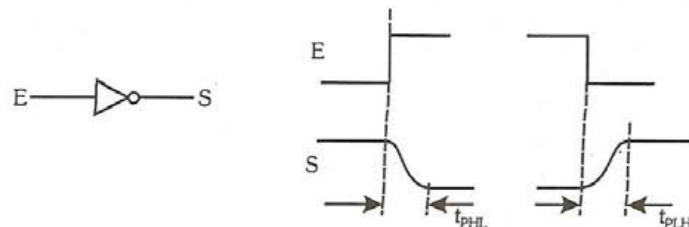



Figura 9.2

Pela figura, considerando que os trechos de sinais da entrada não contenham retardos nas transições de níveis, os de saída, devido a fatores internos do circuito do bloco, irão provocar retardos perceptíveis, acarretando em tempos de atraso.

Esse parâmetro, juntamente com outros, está diretamente relacionado com a **velocidade** de trabalho do bloco lógico, pois, em regime de chaveamento rápido exigido (alta freqüência), será bastante significativo.

9.2.4 Imunidade ao Ruído

Vamos chamar de imunidade ao ruído, a capacidade que os blocos de determinada família lógica possuem de não receber influências parasitas elétricas ou magnéticas, denominadas ruído, típicas dentro dos sistemas eletrônicos ou sob determinadas condições do ambiente em que estão situados. Existem vários tipos de ruídos que podem agir de diversas formas, porém, em se tratando de níveis lógicos, o ruído pode, principalmente pelo aparecimento de níveis espúrios e indesejáveis, fazer o bloco trabalhar na região de nível indefinido, não executando corretamente a função lógica.

As famílias lógicas possuem um parâmetro denominado **margem de imunidade ao ruído**, que determina o quanto de tolerância irá haver sobre os limites dos níveis lógicos, sem que haja alteração na sua funcionalidade. Este parâmetro será uma característica típica da família, sendo um item imprescindível na escolha da tecnologia para a confecção de projetos.

Nos itens relativos às famílias TTL e CMOS, abordaremos de forma numérica, estes parâmetros.

9.3 Blocos Lógicos Estruturados com Diodos

Antes do estudo das famílias lógicas construídas em circuitos integrados, vamos abordar tópicos referentes à estruturação de circuitos de portas lógicas, utilizando como elementos principais os diodos.

Podemos utilizar diodos como chaves e, devidamente conectados, utilizá-los para a construção dos circuitos das portas E e OU. Este tipo de estruturação é ainda encontrado na prática isoladamente em alguns sistemas digitais específicos, sendo vantajosa sua utilização, principalmente nos casos em que se exige a função lógica com níveis de tensão e corrente de saída superiores aos encontrados em circuitos integrados. Convém ressaltar que os circuitos lógicos com diodos não são implementados em circuitos integrados, ficando estes a cargo de outros elementos semicondutores, mais apropriados.

Com diodos, podem ser estruturados apenas os circuitos das portas E e OU, podendo estas operar com níveis de entrada positivos ou com níveis de entrada negativos. Nestes circuitos, na chamada **lógica positiva**, o nível 1 será um valor positivo de tensão (+Vcc), e na **lógica negativa**, o nível 1 será um valor negativo de tensão (-Vcc). Vamos, a seguir, verificar estes circuitos:

A figura 9.3 apresenta o circuito da porta E com terminais de entrada, estruturado para trabalhar em lógica positiva (a), e em lógica negativa (b).

(a) **(b)**

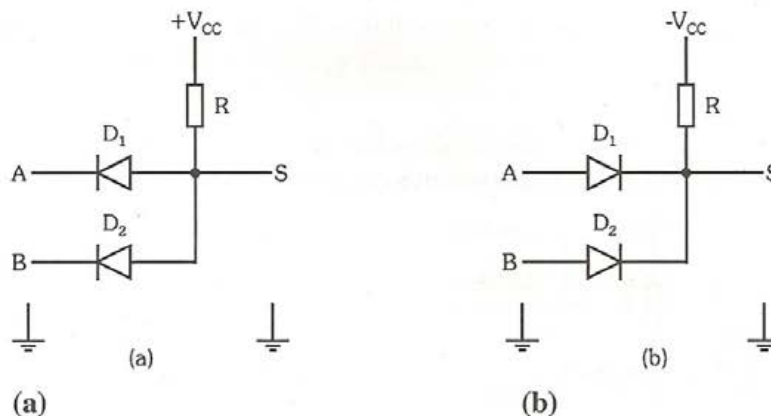



Figura 9.3

Vamos, a seguir, levantar a tabela da verdade de cada circuito verificando o estado do diodos em cada caso:

(a) Porta E de lógica positiva (nível 0 = 0 e nível 1 = +Vcc):

A	B	D_1	D_2	S
0	0	conduz	conduz	0
0	+Vcc	conduz	cortado	0
+Vcc	0	cortado	conduz	0
+Vcc	+Vcc	cortado	cortado	+Vcc

Tabela 9.2

(b) Porta E de lógica negativa (nível 0 = 0 e nível 1 = -Vcc):

A	B	D_1	D_2	S
0	0	conduz	conduz	0
0	-Vcc	conduz	cortado	0
-Vcc	0	cortado	conduz	0
-Vcc	-Vcc	cortado	cortado	-Vcc

Tabela 9.3

Verificando os resultados finais (colunas S), concluímos que os circuitos se comportam como portas E, com saídas compatíveis conforme o tipo de lógica utilizado.

A figura 9.4 apresenta o circuito da porta OU com dois terminais de entrada, estruturado para trabalhar em lógica positiva (a), e em lógica negativa (b).

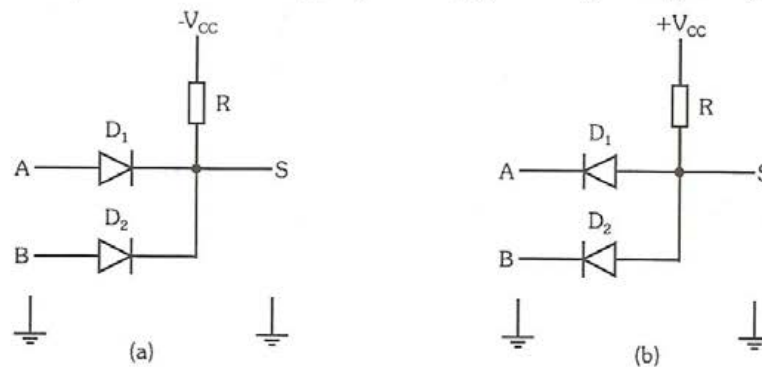



Figura 9.4

Podemos notar que o circuito da porta OU de lógica positiva é idêntico ao da E de lógica negativa. A igualdade também ocorre para o da OU de lógica negativa em relação ao da E de lógica positiva. Através destas observações, concluímos que são apenas dois circuitos, que conforme o tipo de lógica a que estão submetidos, comportam-se como E, ou como OU.

Vamos, da mesma forma, levantar a tabela da verdade de cada um dos circuitos, verificando o estado do diodos em cada caso:

(**a**) Porta OU de lógica positiva (nível 0 = 0 e nível 1 = +Vcc):

A	B	D_1	D_2	S
0	0	conduz	conduz	0
0	+Vcc	cortado	conduz	+Vcc
+Vcc	0	conduz	cortado	+Vcc
+Vcc	+Vcc	conduz	conduz	+Vcc

Tabela 9.4

(**b**) Porta OU de lógica negativa (nível 0 = 0 e nível 1 = -Vcc):

A	B	D_1	D_2	S
0	0	conduz	conduz	0
0	-Vcc	cortado	conduz	-Vcc
-Vcc	0	conduz	cortado	-Vcc
-Vcc	-Vcc	conduz	conduz	-Vcc

Tabela 9.5

Verificando os resultados finais (colunas S), concluímos que os circuitos comportam-se como portas OU , com saídas compatíveis conforme o tipo de lógica utilizado.

O circuito da porta OU visto, em ambas as lógicas, pode ser simplificado para atuar sem a fonte de alimentação, sendo o nível de saída obtido diretamente, a partir dos níveis aplicados às entradas. A figura 9.5 mostra o circuito da porta OU simplificado para atuar na lógica positiva (a), e na negativa (b).

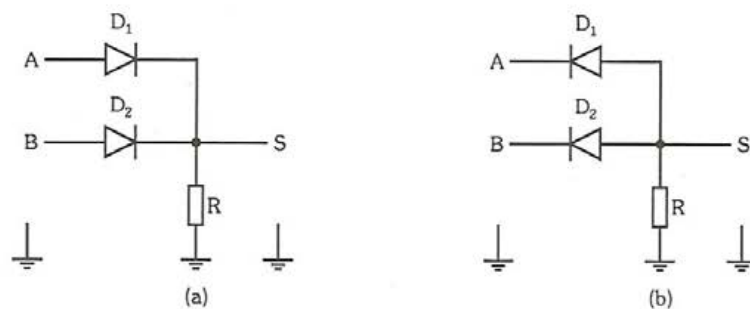



Figura 9.5

No circuito da figura 9.5 (a), ao ser aplicado um nível 1 (+Vcc), a qualquer uma ou em ambas as entradas, o respectivo diodo irá conduzir, fazendo surgir na saída S, devido à queda de tensão em R, um nível 1 positivo compatível com a lógica utilizada. O mesmo irá ocorrer com o circuito da figura 9.5 (b), para o nível 1 na lógica negativa (-Vcc). Assim sendo, estes circuitos atuarão como portas OU, dentro das respectivas lógicas, executando a mesma tabela da verdade.

9.4 Blocos Lógicos Estruturados em Circuitos Integrados

Conforme já citado, na vida prática, existe disponível toda uma série de circuitos lógicos básicos dispostos em circuitos integrados comerciais pertencentes às famílias TTL e CMOS.

Para a construção destes circuitos, a tecnologia TTL utiliza **transistores bipolares**, ou seja, comuns de junção NPN ou PNP. Já a tecnologia CMOS utiliza transistores **MOS-FET (Metal Oxide Semiconductor Field Effect Transistor**) complementares, do tipo N e do tipo P.

A utilização da tecnologia **MOS**, aqui generalizada, apresenta uma série de facilidades, principalmente nos aspectos construtivos dentro dos circuitos integrados, reduzindo de maneira considerável as etapas de integração. Por este motivo, outras modalidades desta tecnologia (**NMOS** e **PMOS**) são utilizadas na implementação de sistemas mais complexos (microprocessadores, memórias de alta capacidade, etc.) de grande quantidade de componentes por chip, chegando hoje em alguns casos, na casa de milhões.

As **escalas de integração**, ou seja, a faixa relativa ao número de componentes por chip, são determinadas pela quantidade de portas ou dispositivos ativos dentro do circuito integrado. Estas escalas recebem uma denominação apropriada conforme o número destes elementos existentes internamente. A tabela 9.6 apresenta as escalas de integração com as respectivas densidades expressas em portas por chip.

Designação	Significado	Densidade (portas por chip)
SSI	*Small Scale Integration*	< 12
MSI	*Medium Scale Integration*	13 a 99
LSI	*Large Scale Integration*	100 a 999
VLSI	*Very Large Scale Integration*	1000 a 99999
ULSI	*Ultra Large Scale Integration*	> 100000

Tabela 9.6

Os circuitos integrados pertencentes às famílias TTL e CMOS enquadram-se nos níveis de integração SSI e MSI; já os outros sistemas mais complexos, anteriormente citados, enquadram-se nos demais níveis.

Para constituir os circuitos internos dos blocos lógicos e assim constituir toda a família lógica, os transistores são dimensionados para atuar como chaves. Vamos, a seguir, estudar o comportamento dos transistores citados, para estruturar os circuitos internos dos blocos lógicos de ambas as famílias.

9.4.1 Transistor Bipolar como Chave

De acordo com a tensão aplicada à base, um transistor bipolar ou comum pode operar no corte ou na saturação, sendo estas duas situações análogas à chave aberta e fechada. O circuito da figura 9.7 mostra a configuração básica de um transistor NPN operando como chave.

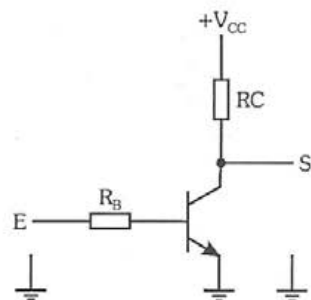



Figura 9.7

As situações de corte e saturação são impostas pela polarização, ou seja, são obtidas em função do correto dimensionamento de R_C e R_B, e pela variação do ponto de trabalho em função da tensão aplicada entre base e emissor do transistor. Para este circuito, o comando da chave será o potencial aplicado à entrada E, ou seja, esta tensão de base.

O transistor comportar-se-á como chave aberta quando aplicarmos um potencial 0 ou negativo na entrada E. Neste caso, operará na situação de corte, pois estaremos aplicando corrente 0 à sua base, sendo a tensão na saída do circuito igual ao potencial da fonte (+Vcc).

Analogamente, o transistor comportar-se-á como chave fechada quando aplicarmos um potencial positivo nesta mesma entrada. Neste caso, operará na saturação e a tensão entre coletor e emissor cairá para 0,3V no máximo, resultando assim em uma baixa tensão (Vs = 0,3V), sendo considerada como nível 0.

A figura 9.8 ilustra as situações explicadas de corte (a) e de saturação (b) do transistor NPN.

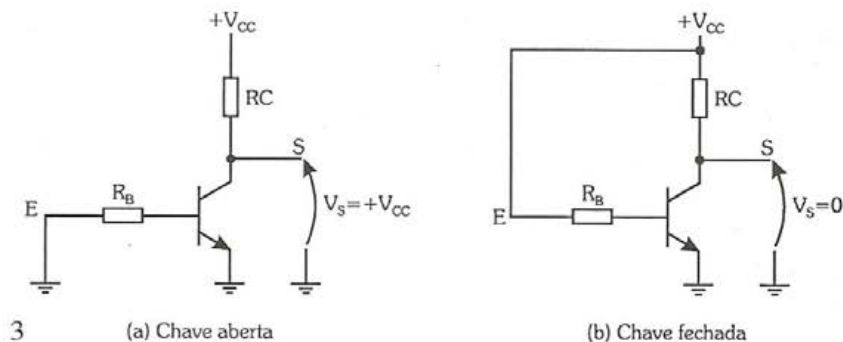



Figura 9.8

Convém observar que atuando nestas situações, o circuito irá se comportar como um inversor, pois se aplicarmos um nível à sua entrada, este aparecerá complementado na saída.

Utilizaremos estes conceitos para explicar o funcionamento de outros circuitos básicos dentro da família TTL.

9.4.2 MOS-FET como Chave

Da mesma forma que o transistor bipolar, um MOS-FET pode, conforme a polarização aplicada, atuar como uma chave aberta ou fechada. O princípio consiste em utilizar um MOS-FET do modo indução, e aplicar uma tensão conveniente, conforme o tipo de transistor (canal N ou canal P), entre porta (gate) e fonte (source), obedecendo à polarização aplicada ao terminal dreno (drain). A figura 9.9 apresenta um MOS-FET do tipo canal N (a), e outro do tipo canal P (b) polarizados para atuarem como chaves.

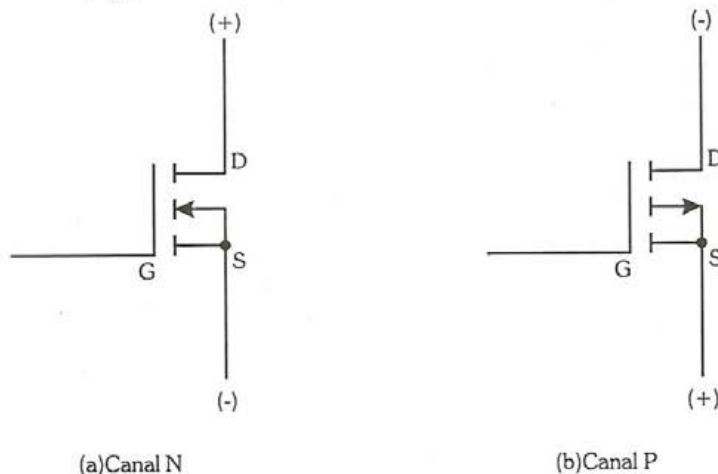



Figura 9.9

Para o MOS-FET do tipo canal N, inicialmente no estado de alta resistência, se aplicarmos no terminal porta (G), um potencial positivo em relação ao terminal fonte (S), este apresentará uma baixa resistência entre D e S, caracterizando uma condução. Neste caso, o MOS-FET comportar-se-á como chave fechada, fazendo o potencial entre D e S cair para um baixo valor de tensão. Se, no entanto, aplicarmos um outro potencial negativo ou nulo, o mesmo se comportará como uma chave aberta, aparecendo entre D e S um alto potencial, ou seja, o da fonte de alimentação ($+V_{DD}$). Para ilustrar, a figura 9.10 apresenta estes casos de polarização colocados.

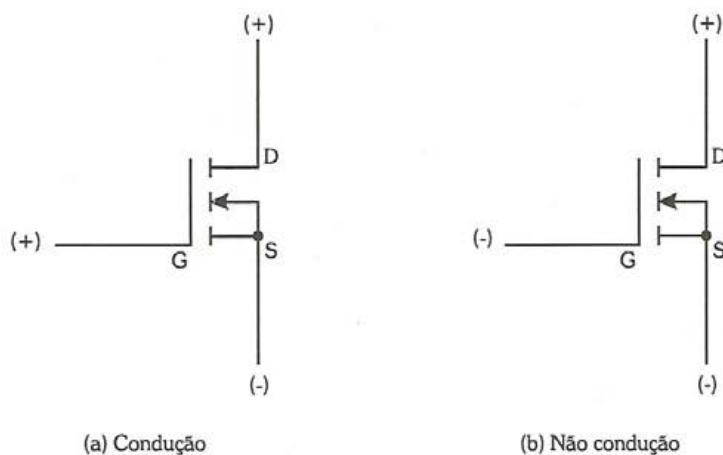



Figura 9.10

Para o MOS-FET do tipo canal P, inicialmente no estado de alta resistência, se aplicarmos no terminal porta (G), um potencial negativo em relação ao terminal fonte (S) ou nulo em relação ao dreno (mesmo potencial), este apresentará uma baixa resistência entre D e S, caracterizando uma condução. Neste caso, o MOS-FET comportar-se-á como chave fechada, fazendo o potencial entre D e S cair para um baixo valor de tensão. Se, no entanto, aplicarmos um outro potencial positivo em relação ao terminal fonte (S) , o mesmo se comportará como uma chave aberta, aparecendo entre D e S um alto potencial, ou seja, o da fonte de alimentação ($-V_{DD}$). Para ilustrar, a figura 9.11 apresenta estes casos de polarização colocados.

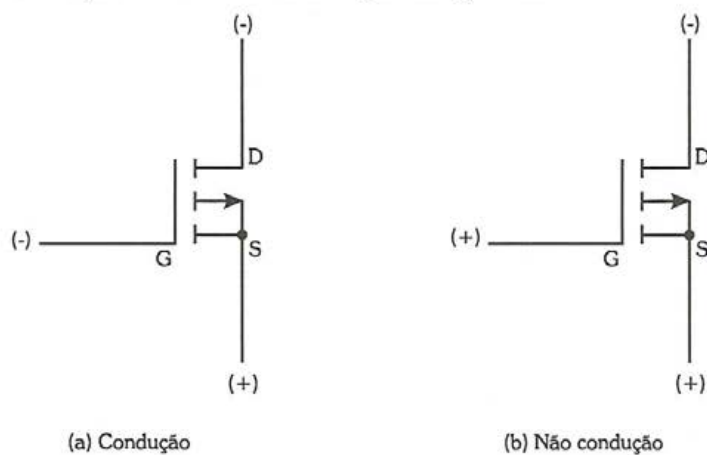



Figura 9.11

Utilizaremos estes conceitos para explicar o funcionamento dos circuitos básicos da família CMOS.

9.5 Família TTL

A família TTL é derivada da antiga família DTL, sendo o resultado de uma série de inovações tecnológicas. Uma delas é a utilização nos seus circuitos internos de transistores bipolares de vários emissores, também conhecidos como **multiemissores**. Trata-se de uma família pioneira, tradicional e muito utilizada ao longo dos anos, devido principalmente ao seu fácil manuseio, e à colocação no mercado de uma série de circuitos integrados comerciais e padronizados.

A seguir, vamos analisar um circuito TTL padrão, que utiliza as ligações no estágio de saída denominadas **Active Pull-Up** (puxar para cima ativo) e **Toten-Pole** (em poste), termos estes de difícil adaptação na tradução. A figura 9.12 apresenta o circuito de uma porta NE TTL.

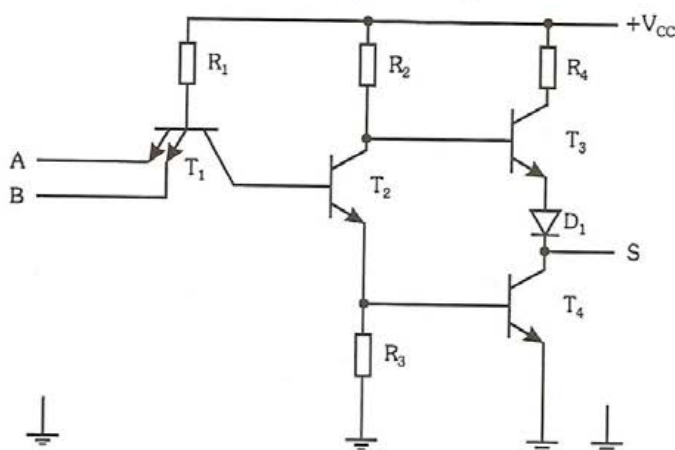



Figura 9.12

A presença de T_3 no coletor caracteriza o Active Pull-Up e sua ligação através do diodo D_1 sobre o coletor de T_4, formando um elevador de potencial, o Toten-Pole.

Quando tivermos uma ou ambas as entradas A e B em nível 0, a respectiva junção base-emissor do transistor T_1 irá conduzir levando T_2 ao corte por ausência de corrente de base. Conseqüentemente, pelo mesmo motivo, T_4 também será levado ao corte. O transistor T_3 , por sua vez, estará

conduzindo, pois, por R_2 fluirá uma corrente através de sua base, e se comportará para a saída como um seguidor de emissor, fazendo aparecer em S um potencial igual a $+V_{CC}$ (nível 1).

Quando ambas as entradas estiverem em aberto ou em nível 1, devido ao corte da junção base-emissor de T_1, por R_1 fluirá uma corrente que irá saturar T_2 e conseqüentemente T_4. Devido à elevação do potencial de base por D_1, o transistor T_3 estará cortado e ocasionará, na saída, um baixo potencial, equivalente a nível 0.

Transpondo estas situações para a tabela da verdade, notaremos que o circuito funcionará como uma porta NE, sendo padrão nesta família.

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela 9.7

Convém ressaltar que, nesta família, a entrada em aberto equivale a nível lógico 1.

Outra observação é que tanto na situação de nível 0 ou 1 de saída, o bloco estará apto a fornecer o nível a um terminal de entrada de outro bloco TTL, sendo respeitados os parâmetros V_{IL} e V_{IH}. Na situação de nível 0 de saída, em especial, a compatibilidade será obtida através da condução de T_4, pois pelo coletor fluirá a corrente proveniente da entrada do bloco seguinte, funcionando T_4 como receptor de corrente, sendo por este motivo denominado transistor **Pull-Down** (puxar para baixo), outro termo de difícil tradução.

9.5.1 Características Gerais e Parâmetros da Família TTL

Os circuitos TTL são produzidos em duas séries comercias: a série *74XXX* e *54XXX*, sendo esta última denominada série militar ou profissional, devido à maior margem de variação nas especificações de alimentação e temperatura, assegurando a confiabilidade no desempenho em condições máximas.

Os valores lidos em manuais são valores dos diversos parâmetros para uma tensão de alimentação de 5V a 25 °C. As especificações da série comum (*74XXX*) devem garantir esse funcionamento com 5% de tolerância numa faixa de temperatura de 0 °C a 70 °C. Já as especificações da série militar (*54XXX*) garantem o funcionamento com 10% de tolerância numa faixa de temperatura de -55° a 125 °C.

Vamos agora, enumerar os principais parâmetros encontrados nos manuais em nomenclaturas originais:

1- Alimentação (Vcc): Na família TTL, temos para todos os blocos uma alimentação de 5V. Para a série 54 temos Vcc mínimo = 4,5V e Vcc máximo = 5,5V que são valores dentro da especificação militar de 10% de tolerância. Para a série de 74, temos Vcc mínimo = 4,75V e Vcc máximo = 5,25V que são valores dentro da especificação comum de 5% de tolerância.

2- Níveis de entrada e saída, para a versão padrão (**TTL Standard**):

TTL *Standard*		
Parâmetros	**Valores**	**Unidade**
V_{IL}	0,8	V
V_{OL}	0,4	V
V_{IH}	2,0	V
V_{OH}	2,4	V
I_{OL}	16	mA
I_{IL}	1,6	mA
I_{OH}	400	μA
I_{IH}	40	μA

Tabela 9.8

3- Fan-out: Na versão padrão, o Fan-Out é igual a 10 (ver exemplo de cálculo no item 9.2.2), ou seja, podemos ligar à saída deste bloco no máximo outros 10 blocos similares. Este valor, normalmente, é generalizado para toda a família TTL.

4- Tempo de atraso de propagação: Este parâmetro varia conforme a versão utilizada, sendo o valor médio aproximado da ordem de 10ns na versão mais comum. A tabela 9.9 apresenta os tempos de atrasos típicos de subida t_{PLH} (Low to High), e de descida t_{PHL} (High to Low), para esta versão.

TTL	*Standard*	
Parâmetros	**V. típico**	**Unidade**
t_{PLH}	11	ns
t_{PHL}	7	ns

Tabela 9.9

5- Imunidade ao Ruído: A margem de imunidade ao ruído especificada para a família TTL de maneira geral, é obtida supondo a ligação da saída de um bloco para a entrada de outro, sendo definida por aquela margem de segurança colocada pelo fabricante entre os parâmetros de entrada e saída. É calculada pela diferença dos parâmetros relativos a esses níveis de tensão (margem de imunidade ao ruído DC). Assim sendo, temos:

No nível 1: $\Delta V_{RH} = V_{OH}\,(\text{mín.}) - V_{IH}\,(\text{mín.}) = 2{,}4 - 2{,}0 = 0{,}4V$

No nível 0 : $\Delta V_{RL} = V_{IL}\,(\text{máx.}) - V_{OL}\,(\text{máx.}) = 0{,}8 - 0{,}4 = 0{,}4V$

$\therefore \Delta V_R = 0{,}4V$

A margem de imunidade ao ruído para a família TTL, de maneira geral, é igual a 0,4V, e é considerada baixa em relação à família CMOS, sendo os componentes CMOS mais apropriados para operar frente a situações de alto nível de ruído.

6- Potência Dissipada: O consumo médio de potência da família TTL é da ordem de 10mW por porta na versão mais comum.

9.5.2 Tipos de Blocos da Família TTL

A família TTL, através de suas séries, colocou blocos disponíveis no mercado de componentes, com muitas possibilidades. Entre eles, vamos destacar os blocos **open-collector, tri-state** e **schimitt-trigger**.

9.5.2.1 Open-collector

A família TTL possui blocos lógicos com construção em open-collector (coletor aberto). Os circuitos destes blocos são semelhantes aos dos blocos convencionais, com a única diferença de não ter internamente o resistor de coletor ligado ao +Vcc. Este deve ser ligado externamente quando da utilização do bloco. O circuito interno de uma porta NE TTL open-collector é visto na figura 9.13.

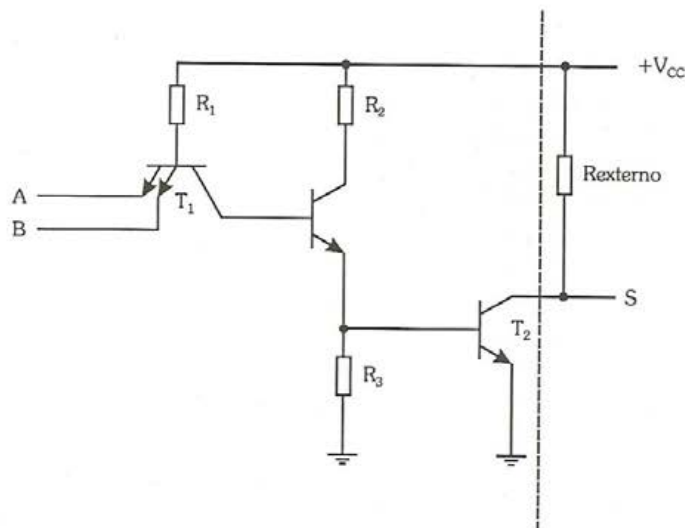



Figura 9.13

Esta configuração permite o controle externo da corrente de coletor, proporcionando inclusive o aumento do Fan-out. Além disso, permite a ligação conjunta de várias saídas através de um único resistor de coletor, formando uma ligação denominada **E por fio (Wired-And)**, pois executa a função de uma porta E, apenas com a ligação. A figura 9.14 mostra a ligação, a tabela e a simbologia utilizadas para se obter uma função E através de blocos open--collector.

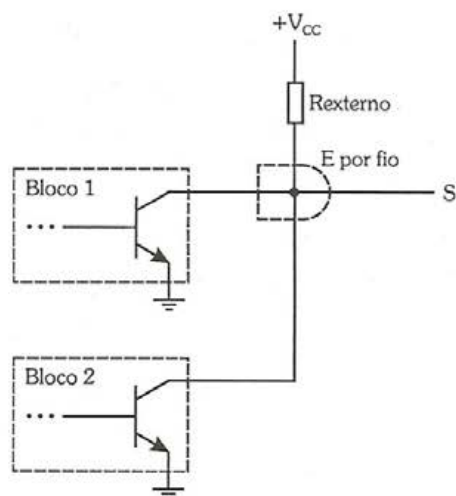



Figura 9.14

S_1	S_2	S
0	0	0
0	1	0
1	0	0
1	1	1

Tabela 9.10

Notamos pela figura, que o nível 0 é obtido pela saturação de cada transistor ou por ambos, conforme a função lógica de cada bloco, sendo a respectiva corrente de coletor fornecida pelo mesmo resistor colocado externamente. O nível 1 é obtido pelo corte de ambos.

Como outra aplicação, podemos citar o uso muito comum de saídas open-collector para ativar displays de 7 segmentos a led, possibilitando o controle de luminosidade pelo resistor de coletor calculado e colocado externamente.

9.5.2.2 Tri-state

Conforme já visto, existem blocos que apresentam um terceiro estado de saída (tri-state) caracterizado por uma alta impedância.

Para ativar o tri-state (também utilizados: **3-state** ou **three-state**), o bloco específico possui um terminal que, conforme o nível lógico assumido, faz a saída permanecer ou não em alta impedância. Para ilustrar esta possibilidade a figura 9.15 mostra o circuito simplificado de uma porta NE TTL de duas entradas com saída tri-state.

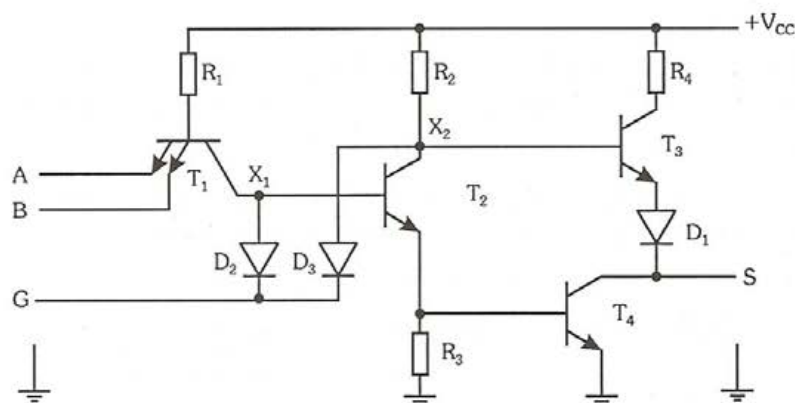



Figura 9.15

No circuito, se aplicarmos nível 1 ao terminal de entrada de controle da saída (G) ou o deixarmos em aberto, o circuito funcionará normalmente como uma porta NE, pois D_2 e D_3 estarão cortados. Se, no entanto, aplicarmos nível 0, devido à respectiva condução de corrente pelos mesmos diodos, os pontos X_1 e X_2 cairão para baixos potenciais, levando T_2 , T_3 e T_4 para a situação de corte. O terminal de saída, neste caso, será praticamente desligado do circuito, ocasionando o estado de alta impedância.

Na família TTL, as saídas tri-state são encontradas fazendo parte de vários dispositivos, porém, isoladamente como portas, estão disponíveis apenas em buffers comuns e inversores.

As aplicações de dispositivos com saídas tri-state são muitas, principalmente em sistemas com microprocessadores, onde vários circuitos integrados utilizam o mesmo conjunto de fios de forma compartilhada, formando assim a já conhecida via de dados do sistema.

9.5.2.3 Schimitt-Trigger

São também encontrados disponíveis na família TTL, blocos configurados com entradas Schimitt-Trigger (gatilho de Schimitt). Este tipo de bloco possibilita tornar rápidas, as variações lentas dos níveis de tensão de determinados sinais aplicados à sua entrada, causando na saída o aparecimento de uma onda quadrada bem-definida. Em outras palavras, este tipo de bloco, além de realizar sua função lógica, quadra o sinal aplicado à entrada, desde que sejam respeitados os parâmetros mínimos e máximos de tensão especificados para o bloco.

O bloco irá considerar iguais a 0, os valores de entrada abaixo do especificado por **V_T - (Negative-Going Threshold Voltage)** ou limiar negativo de tensão, e irá considerar iguais a 1, os valores acima de **V_T + (Positive-Going Threshold Voltage)** ou limiar positivo de tensão.

Para ilustrar, a figura 9.16 apresenta um inversor TTL schimitt-trigger (a) e a ação sobre um sinal de variação lenta aplicado à sua entrada (b).

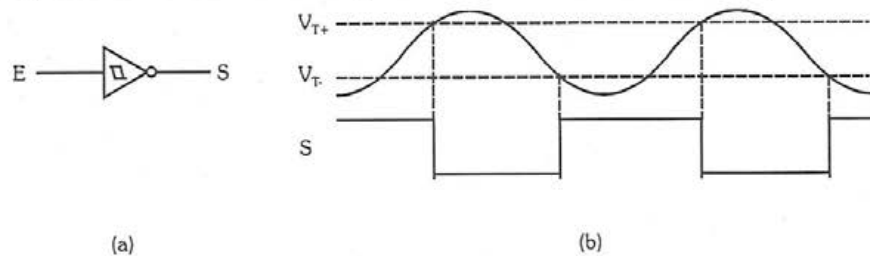



Figura 9.16

O símbolo (**histerese**) presente no inversor é utilizado em manuais de fabricantes para identificar as portas que executam a função de schimitt--trigger, sendo atribuído devido à aparência da característica de transferência do bloco. Para exemplificar esta curva e os valores práticos dos parâmetros V_T - e V_T + , a figura 9.17 mostra a característica de transferência típica do circuito integrado TTL 7414 (6 inversores schimitt-trigger).

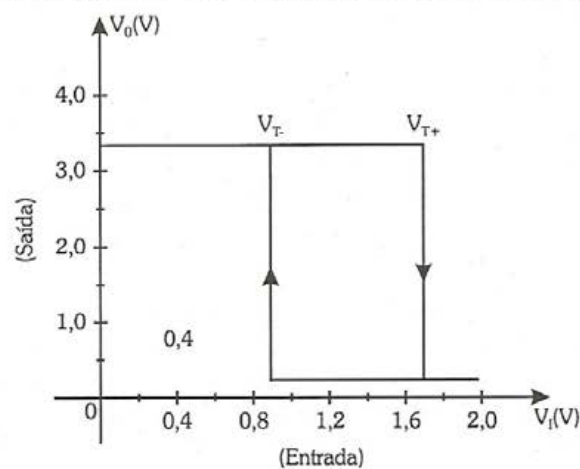



Figura 9.17

Pelo gráfico, notamos que para a saída assumir nível 0 (V_{OL} = 0,2V), é necessário que a variação de entrada atinja aproximadamente V_T+ = 1,7V, e que para assumir nível 1 (V_{OH} = 3,4V), é necessário que a variação de entrada caia abaixo de V_T- = 0,9 V aproximadamente.

Os dispositivos schimitt-trigger são largamente utilizados em sistemas digitais, para transformar em onda quadrada as variações oriundas de sistemas analógicos diversos não compatíveis. Uma dessas aplicações consiste em a partir de uma amostra da tensão senoidal da rede elétrica, obter o sinal de clock quadrado de 60 Hz para, após dividido, fornecer 1 Hz aos contadores de segundos dos relógios digitais.

Além de inversores, são encontradas disponíveis em circuitos integrados da família TTL, portas NE schimitt-trigger.

9.5.3 Versões dos Circuitos TTL

Além dos blocos comuns (Standard), a família TTL possui outras versões de circuitos com a finalidade de atender a solicitações de ordem prática nos parâmetros relativos à velocidade e consumo de potência. A tabela 9.11 apresenta um quadro comparativo entre estas versões e as respectivas identificações.

Versão	Identificação da série	Tempo de atraso de propagação típico por porta	Consumo de potência por porta	Freqüência de clock máxima para flip-flop	Observações
Standard	54/74	10 ns	10 mW	35 MHz	comum
Low power	54L/74L	33 ns	1 mW	3 MHz	baixíssimo consumo
High speed	54H/74H	6 ns	22mW	50 MHz	alta velocidade
Schottky	54S/74S	3 ns	19mW	125 MHz	altíssima velocidade

Versão	Identificação da série	Tempo de atraso de propagação típico por porta	Consumo de potência por porta	Freqüência de clock máxima para flip-flop	Observações
Advanced Schottky	54AS/74AS	1,5 ns	8,5 mW	200 MHz	altíssima velocidade e baixo consumo
Low power Schottky	54LS/74LS	10 ns	2 mW	45 MHz	baixíssimo consumo
Advanced Low power Schottky	54ALS/74ALS	4 ns	1 mW	70 MHz	altíssima velocidade e baixíssimo consumo

Tabela 9.11

Os valores da tabela são válidos para circuitos integrados de portas NE e servem apenas para comparações entre as versões, sendo estimados a partir das faixas disponíveis nos manuais comerciais de diversos fabricantes.

O quadro possibilita a comparação em termos de velocidade e consumo de potência, tomando como ponto de referência a versão comum, seguida da versão de baixo consumo (L) e de alta velocidade (H). Essas versões são diferentes entre si devido a alterações introduzidas nos circuitos e nos valores de seus componentes internos. A partir daí, para as versões do quadro os circuitos apresentam variações sobre a tecnologia Schottky.

A versão Schottky utiliza em seus circuitos o diodo Schottky, elemento semicondutor construído com metal de um lado da junção interna para aumentar a velocidade de comutação, que devidamente colocado entre base e coletor de um transistor, forma um conjunto denominado **Transistor Schottky**. Este conjunto, quando utilizado para chaveamento, não atinge a saturação totalmente devido à ligação, apresentando um tempo de comutação extremamente baixo e consequentemente uma altíssima velocidade de trabalho. A figura 9.18 mostra a ligação de um diodo Schottky em um transistor bipolar para formar o referido conjunto (a) e a simbologia utilizada para este (b).

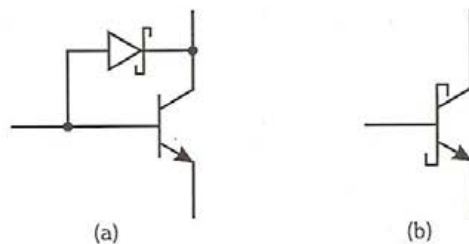
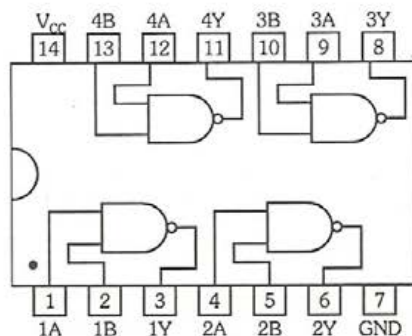



Figura 9.18

As alterações introduzidas nos circuitos e nos valores dos componentes, produziram em relação à versão Schottky (S), uma variação de menor consumo (LS), e nas versões Schottky Avançadas (AS e ALS), uma grande melhora no desempenho total, principalmente no produto velocidade-consumo, constituindo-se nos menores entre todas as versões existentes.

9.5.4 Circuitos Integrados TTL

A família TTL colocou no mercado uma série de circuitos integrados padronizados com configurações de pinagens disponíveis nos manuais dos fabricantes. São circuitos integrados de 14 pinos ou mais, conforme a complexidade do circuito agregado, com encapsulamentos denominados **DIP (Dual-In-Line Package)**, cuja identificação da disposição dos terminais se faz através da vista superior, em sentido anti-horário, a partir do ponto de referência colocado no pino 1, próximo ao chanfro existente no bloco. Para exemplificar, a figura 9.19 apresenta a pinagem do circuito integrado 7400 (4 NE com 2 entradas), sendo esta válida também para o 5400 e, ainda, para as versões 74L00, 74H00, 74S00, 74AS00, 74LS00, 74ALS00.

Alimentação:

Pino 14: +Vcc

Pino 7: terra ou GND (ground)

Figura 9.19

9.6 Família CMOS

A outra família de extrema importância a ser abordada é a CMOS (Complementary MOS). Trata-se de uma família que tem seus circuitos construídos por transistores MOS-FET complementares do tipo canal N e canal P. Suas configurações básicas permitem obter-se uma série de vantagens, tais como: alto Fan-Out, alta margem de imunidade ao ruído e baixíssimo consumo, sendo esta uma de suas mais importantes características.

Vamos, a seguir, analisar o funcionamento dos blocos lógicos principais desta família que são as portas NOU e NE. A figura 9.20 mostra o circuito básico de uma porta NOU CMOS.

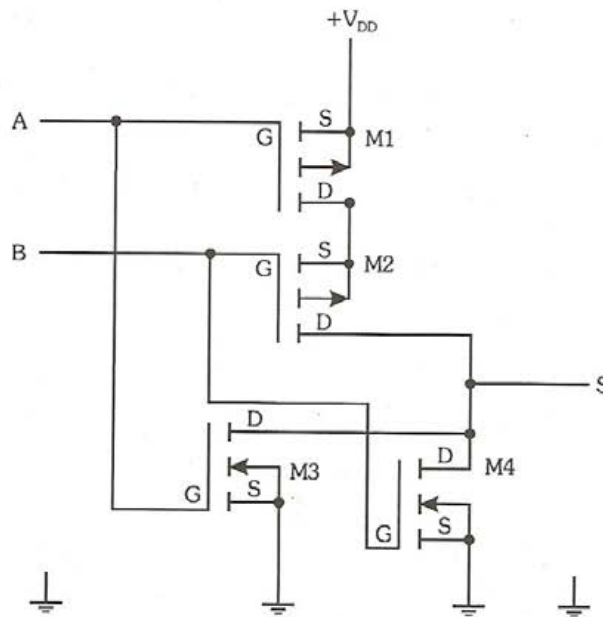



Figura 9.20

Quando ambas as entradas estiverem em 0 (potencial do terra), os MOS-FET canal P, M_1 e M_2 estarão conduzindo e os MOS-FET canal N, M_3 e M_4 estarão cortados. Isso fará com que a tensão de saída assuma valor igual a $+V_{DD}$ (nível 1). Quando pelo menos uma das entradas estiver em $+V_{DD}$ (nível 1), teremos o respectivo MOS-FET canal N, M_3 ou M_4 conduzindo, fazendo com que na saída tenhamos uma tensão igual a 0. Transpondo estas situações para uma tabela verdade, concluímos que o circuito comporta-se como uma porta NOU:

A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

Tabela 9.12

Vamos analisar agora, o funcionamento da porta NE CMOS. O circuito básico é visto na figura 9.21.

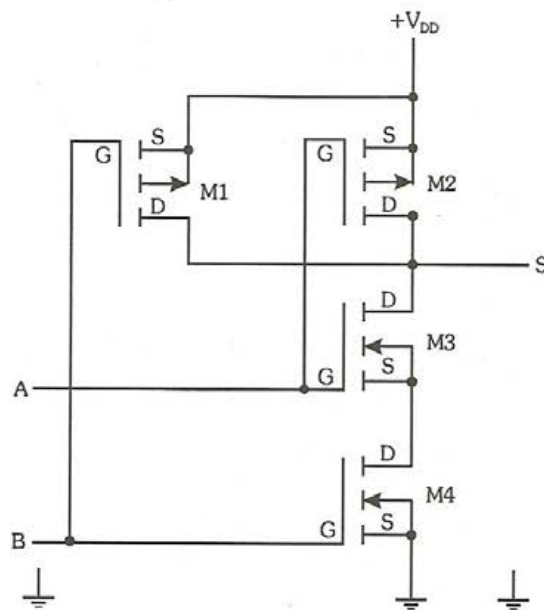



Figura 9.21

Quando pelo menos uma das entradas estiver em 0, o respectivo MOS-FET canal N, M_3 ou M_4 estará cortado e o respectivo MOS-FET canal P, M_1 ou M_2 estará conduzindo, logo, teremos na saída uma tensão igual a VDD (nível 1). Quando ambas as entradas estiverem em +VDD (nível 1), tanto M_3 como M_4 estarão conduzindo, ficando M_1 e M_2 cortados, logo, teremos na saída uma tensão igual a 0. Transpondo estas situações para uma tabela verdade, concluímos que o circuito comporta-se como uma porta NE:

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela 9.13

Convém ressaltar que a partir destes circuitos básicos o fabricante pode estruturar, internamente no circuito integrado, qualquer outro bloco mantendo as mesmas características de entrada e saída.

Um outro ponto importante a ser ressaltado é que ao contrário da família TTL, não é aconselhável deixar terminais de entrada em vazio nas portas CMOS, pois nesta situação, estes se tornam susceptíveis à captação de cargas estáticas e ruídos indesejáveis, causando pela polarização errônea dos dispositivos, um aumento da dissipação de potência e conseqüente sobreaquecimento. Os terminais não utilizados devem ser conectados, conforme o caso da função lógica envolvida, ao terra ou V_{DD} do circuito.

9.6.1 Características Gerais e Parâmetros da Família CMOS

A família CMOS possui circuitos integrados disponíveis nas séries comercias *4000A*, *4000B* e *54/74C,* sendo esta última semelhante à TTL na pinagem dos circuitos integrados e função dos blocos disponíveis. Além destas, a família CMOS também possui versões de alta velocidade e melhor desempenho: 74HC/74HCT (**High-speed CMOS**), sendo a HCT especialmente desenvolvida para atuar com parâmetros de tensões compatíveis com TTL-LS, e as apropriadas para operar com baixa tensão de alimentação: 74LV/74LVC (**Low Voltage CMOS**).

Os circuitos integrados CMOS são dimensionados para operar na faixa de temperatura de -40° a +85 °C nas séries comuns, e nas variações de uso profissional (militar) na faixa de -55° a +125 °C.

Vamos agora, enumerar os principais parâmetros encontrados nos manuais em nomenclaturas originais:

1- Alimentação (V_{DD}): Quanto à tensão de alimentação, esta família permite para as séries 4000 e 74C operarem na faixa de 3V a 15V, para a versão HC de 2V a 6V e para a HCT de 4,5V a 5,5V. Para as séries de baixa voltagem, a faixa de 1V a 3,6V para a LV e 1,2V a 3,6V para a LVC, sendo estas especialmente projetadas para operar com 3,3V, tensão típica de vários sistemas atuais.

Podemos notar que esta família e suas versões apresentam a vantagem de possuir uma larga faixa de tensão de alimentação, não necessitando de regulagem precisa na fonte como no caso da TTL.

2- Níveis de tensões e correntes de entrada e saída: Os blocos da família CMOS apresentam estes níveis, especificados nos manuais, com variações em função da versão e tipo de bloco utilizado. De maneira geral, apresentam nas entradas, valores de V_{IL} (máx.) iguais a 30% do V_{DD} e V_{IH} (min.) iguais a 70% do V_{DD} , com exceção da versão HCT que possui estes níveis iguais a TTL-LS. Nas saídas dos blocos, devido principalmente à baixa absorção de corrente na ligação com o bloco seguinte (alta resistência de entrada), apresentam valores muito próximos a 0 (V_{OL} máx.) e V_{DD} (V_{OH} min.). A tabela 9.14 apresenta os valores de tensões e correntes para a série 4000B, operando com V_{DD} igual a 5V.

CMOS *4000B*		
Parâmetros	**Valores**	**Unidade**
V_{IL}	1,5	V
V_{OL}	0,05	V
V_{IH}	3,5	V
V_{OH}	4,95	V
I_{OL}	0,4	mA
I_{IL}	1	µA
I_{OH}	0,4	mA
I_{IH}	1	µA

Tabela 9.14

3- Fan-Out: Nesta família, de modo generalizado, o Fan-Out é igual a 50, porém varia conforme as versões empregadas. Este valor considerável é devido principalmente à pouca deriva da corrente de saída, em função da alta resistência de entrada dos dispositivos CMOS conectados, sendo a limitação causada pela ação das capacitâncias de entrada dos blocos subseqüentes somados. Devido à compatibilidade de algumas versões com TTL, é comum nos manuais, encontrar este parâmetro definido para um carregamento da saída com TTL-LS, sendo este um menor valor (Fan-Out = 10 para HC/HCT).

4- Tempo de atraso de propagação: Nas séries mais comuns, o tempo de atraso de propagação médio é da ordem de 90ns, constituindo-se em uma grande desvantagem. O problema foi superado com o aparecimento das versões apropriadas para uso em alta velocidade (HC/HCT), com parâmetros compatíveis com os das versões TTL para a mesma finalidade. Para exemplificar, a tabela 9.15 apresenta os parâmetros de velocidade para a série básica e as versões citadas, com tensão de alimentação igual a 5V.

Versão	Tempo de atraso de propagação típico por porta	Freqüência de clock máxima para flip-flop
4000B	90 ns	12 MH_Z
HC/HCT	8 ns	55 MH_Z

Tabela 9.15

5- Imunidade ao Ruído: A margem de imunidade ao ruído para a família CMOS é igual a 45% de V_{DD}, sendo muito alta se comparada com a família TTL. Devido a isso, estes blocos são adequados para ser utilizados em circuitos que operam em sistemas ou ambientes de alto nível de ruído.

6- Potência Dissipada: O consumo de potência da família CMOS (com V_{DD} = 5V) é da ordem de 1nW por porta na série 4000 e 2,5nW por porta na versão 74HC, sendo estes valores muito baixos, caracterizando-se em mais uma grande vantagem desta família.

7- Manuseio: A família CMOS, ao contrário da TTL, possui problemas com o manuseio dos circuitos integrados que devido à ação da

eletricidade de estática, provoca a degradação das junções internas dos chips, comprometendo sua vida útil. A danificação total do bloco pode só acontecer após um certo tempo de uso, causando sérios transtornos ao fabricante do sistema no qual o componente está engajado.

Para contornar o problema, possibilitando um manuseio mais seguro, existe no mercado uma série de dispositivos antiestáticos (pulseiras de aterramento, pisos, borrachas de bancada, estações de solda, etc.), sendo inclusive os circuitos integrados comercializados em embalagens com isolação apropriada.

As versões mais recentes desta família possuem internamente nas entradas e saídas dos blocos, diodos de proteção para evitar a ação da eletricidade estática, porém, aconselha-se seguir da mesma forma as normas de manuseio apropriadas.

9.6.2 Circuitos Integrados CMOS

Da mesma forma que na TTL, a família CMOS colocou no mercado uma série de circuitos integrados padronizados com configurações de pinagens disponíveis nos manuais dos fabricantes. Para exemplificar, a figura 9.22 apresenta a pinagem do circuito integrado 4001B (4 NOU com 2 entradas) e do 74HC04/74HCT04 (6 inversores), sendo estes últimos de mesma pinagem que o 7404 da família TTL.

(a) 4001B

(b) 74HC04/74HCT04

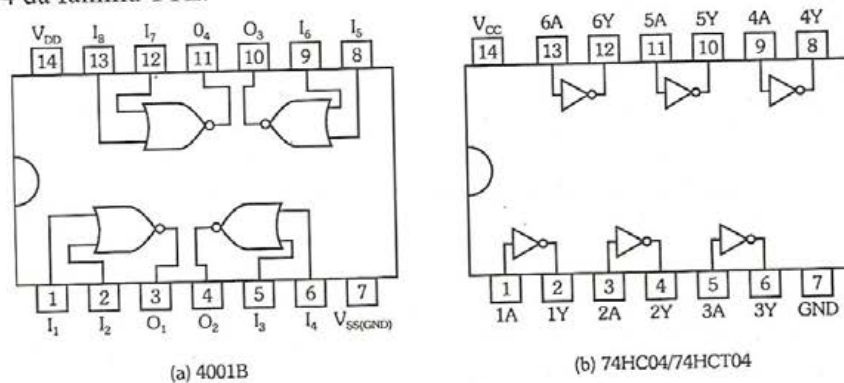



Figura 9.22

9.7 Exercícios Resolvidos

1- Desenhe o circuito que executa a expressão, utilizando diodos com lógica positiva: $S = \overline{A}\,B + A\,\overline{B}$.

Para desenhar este circuito, para facilitar, vamos utilizar o formato matricial, ou seja, desenhar as portas lógicas utilizando linhas e colunas formando um circuito denominado **matriz de diodos**. A entrada de dados será feita por fios de cada variável e os respectivos complementares, pois não é possível formar inversores com diodos. A figura 9.23 apresenta o circuito com esta esquematização.

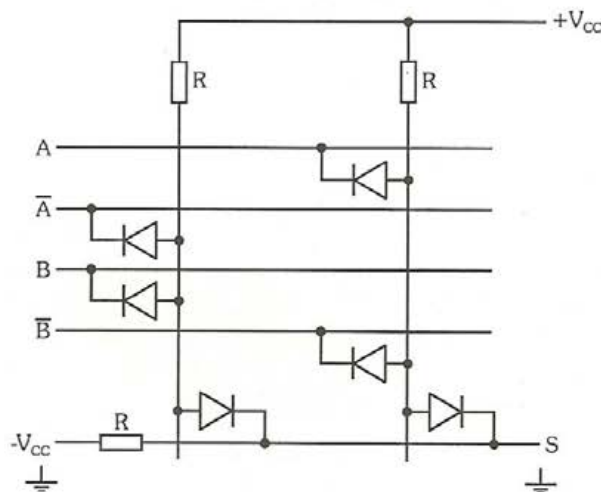



Figura 9.23

Notamos pelo circuito que as linhas verticais (colunas) são relativas às saídas das portas E e a última linha horizontal é a saída geral do circuito, relativa à porta OU.

Para o circuito da OU, poderia ser utilizado o modo simplificado visto na figura 9.5, utilizando a ligação do resistor ao terminal de terra, ao invés de -Vcc. Esta ligação propiciaria a utilização de uma fonte simples ao invés de uma simétrica, como é o caso.

2- Supondo que o circuito da porta lógica, visto na figura 9.24, esteja dimensionado para os transistores atuarem nas situações de corte e saturação, levante sua tabela da verdade e determine a função lógica, utilizando como nível 1 o potencial de +Vcc.

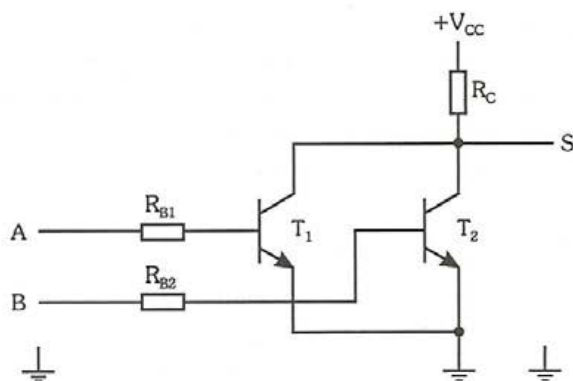



Figura 9.24

No circuito, se uma das entradas A ou B, ou ainda ambas, estiver em nível 1 (+Vcc), o respectivo transistor irá saturar, fazendo com que a saída S apresente um potencial de saturação (0,3V) que representa um nível 0. Se ambas as entradas estiverem em nível 0, teremos os dois transistores cortados e, conseqüentemente, teremos a saída S igual a +Vcc, ou seja, em nível 1. Transpondo estas situações para a tabela da verdade, temos:

A	B	T_1	T_2	S
0	0	cortado	cortado	+Vcc
0	+Vcc	cortado	saturado	0
+Vcc	0	saturado	cortado	0
+Vcc	+Vcc	saturado	saturado	0

Tabela 9.16

Notamos, pela tabela da verdade, que este circuito terá o comportamento de uma função lógica NOU.

3- Para o circuito da figura 9.25, a partir do forma de onda aplicada à entrada E, determine a forma de onda de saída, sabendo-se que as portas pertencem à versão TTL Standard, com um tempo de atraso de propagação igual a 10 ns.

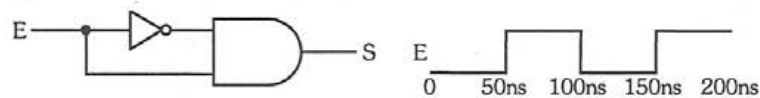



Figura 9.25

Este circuito, teoricamente, se considerássemos os blocos como ideais, sem retardo na resposta, deveria apresentar um nível de saída sempre igual a 0, porém devido ao tempo de atraso de propagação existente nos blocos práticos, irá apresentar um pulso estreito de tensão. A figura 9.26 apresenta a forma de onda de saída obtida a partir da aplicada à entrada.

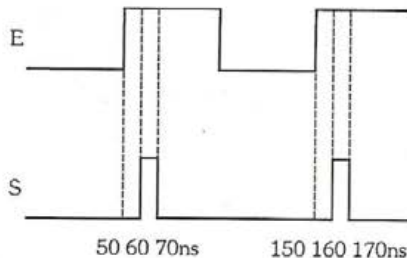



Figura 9.26

Seguindo o sinal da entrada E, ao ser aplicado um nível 0 a esta, o inversor passa a sua saída para 1 após 10 ns de resposta, porém devido ao mesmo nível também ser aplicado à porta E, a saída do circuito irá permanecer no nível 0. Logo após, na transição de E para nível 1, tanto a porta E como o inversor irão recebê-la simultaneamente, passando a saída, após 10 ns, para nível 1 em função da lógica da porta E com o nível 1 de saída do inversor presente durante esta resposta. A partir daí com a descida da saída do inversor para 0, a porta E levará a saída para 0 decorridos os 10 ns de resposta. Assim sendo, o processo se repete para outros ciclos do mesmo sinal, gerando a partir da onda quadrada de entrada, uma série de pulsos estreitos e repetitivos.

Este circuito pode ser utilizado na prática, como gerador de pulsos rápidos de disparo em outros blocos digitais e sistemas derivados.

4 - Utilizando os valores especificados no manual CMOS para o circuito integrado 40106B (6 inversores Schmitt-Trigger), esboce a forma de onda de saída para o sinal visto na figura 9.27, aplicado à entrada de um dos inversores.

Valores para V_{DD} = 5V: V_P = 3,0V (limiar positivo) e V_N = 2,2V (limiar negativo).

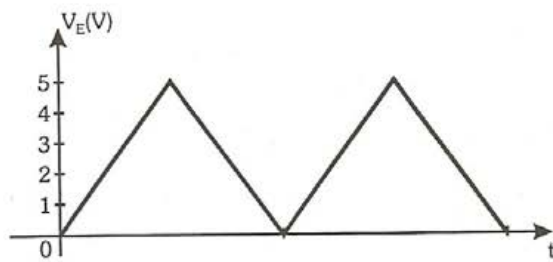



Figura 9.27

O sinal de saída será obtido devido à ação do inversor e dos parâmetros delimitadores dos níveis de tensão especificados. Assim sendo, a mudança do nível de saída para 0 dar-se-á apenas quando a transição de subida do sinal de entrada ultrapassar a V_P, e para nível 1 apenas quando a transição de descida cair abaixo de V_N. A figura 9.28 mostra o sinal de entrada com os parâmetros assinalados e o sinal de saída obtido.

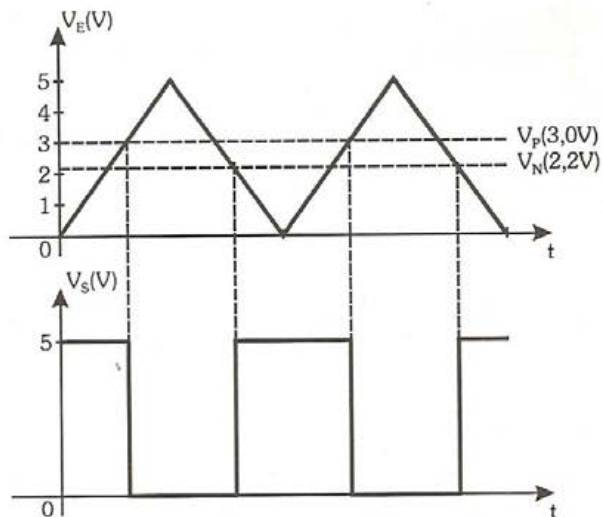



Figura 9.28

Pelo sinal de onda quadrada obtida, notamos que o período e consequentemente a freqüência do sinal triangular foram mantidos, sendo esta uma importante característica dos blocos com esta função.

9.8 Exercícios Propostos

9.8.1- Desenhe o circuito que executa a expressão, utilizando matriz de diodos com lógica positiva: S= A ⊙ B.

9.8.2- Idem ao anterior para a expressão $S = AB\overline{C} + \overline{A}\,\overline{B}C$, com lógica negativa.

9.8.3- Idem ao anterior para $S = (A+B+C).(\overline{A}+\overline{B}+\overline{C})$, com lógica positiva.

9.8.4- Levante a tabela da verdade e determine a função da porta lógica vista na figura 9.29, supondo que o circuito esteja dimensionado para os transistores atuarem nas situações de corte e saturação. Utilize como nível 1 o potencial de +Vcc.

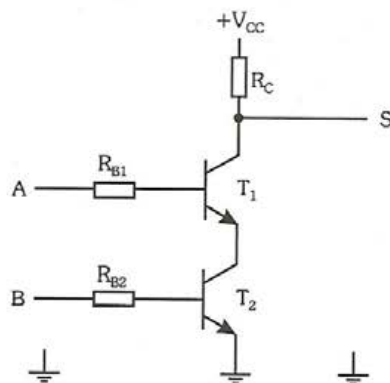



Figura 9.29

9.8.5- Idem ao anterior para o circuito da figura 9.30

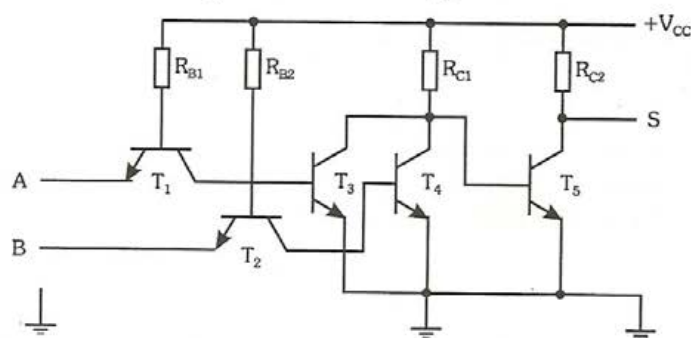



Figura 9.30

9.8.6 - Da mesma forma que os anteriores, levante a tabela da verdade e determine a função do circuito visto na figura 9.31.

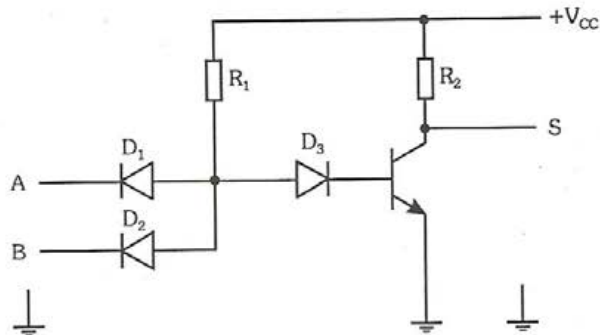



Figura 9.31

9.8.7- No exercício anterior, explique a finalidade do diodo D_3.

9.8.8 - Levante a tabela da verdade do inversor CMOS visto na figura 9.32, mostrando as situações dos transistores M_1 e M_2.

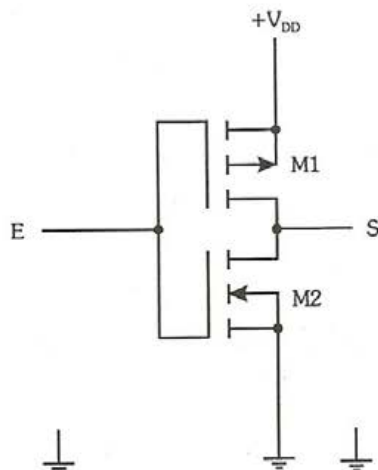



Figura 9.32

9.8.9 - Utilizando os valores especificados no manual TTL, vistos na tabela 9.17, calcule os valores do Fan-Out de nível 0 e 1, para cada versão apresentada na tabela.

TTL	*S*	*AS*	*LS*	*ALS*
I_{OL}	20 mA	20 mA	8 mA	8 mA
I_{IL}	2 mA	500 μA	400 μA	100 μA
I_{OH}	1 mA	2 mA	400 μA	400 μA
I_{IH}	50 μA	20 μA	20 μA	20 μA

Tabela 9.17

9.8.10- Utilizando valores da tabela 9.17 e da tabela 9.1, calcule os valores do Fan-Out de nível 0 e 1, para o caso de utilização da versão Standard com carregamento de saída TTL-LS.

9.8.11- O que representa a faixa compreendida entre V_{IL} máximo e V_{IH} mínimo? E a compreendida entre V_{OL} máximo e V_{OH} mínimo ?

9.8.12- Para o circuito da figura 9.33, a partir da forma de onda aplicada à entrada E, determine a forma de onda de saída, sabendo-se que as portas pertencem à versão TTL-LS, com um tempo de atraso de propagação igual a 10 ns.

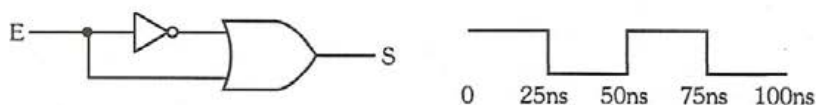



Figura 9.33

9.8.13- Utilizando os valores especificados no manual TTL para o circuito integrado 74LS14 (6 inversores Schmitt-Trigger), esboce a forma de onda de saída para o sinal visto na figura 9.34, aplicado à entrada de um dos inversores, considerando $V_{OL} = 0$ e $V_{OH} = Vcc$. Calcule, ainda, a freqüência do sinal obtido.

Valores: $V_T- = 0,8V$ e $V_T+ = 1,6V$.

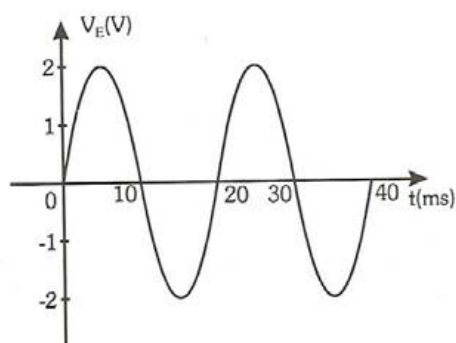



Figura 9.34

9.8.14- Complete a tabela 9.18, estabelecendo uma avaliação comparativa entre os blocos TTL-Standard e CMOS da série 4000B.

FAMÍLIA	Tensão de Alimentação	Potência Dissipada	Margem de Imunidade ao Ruído	Tempo de Atraso de Propagação
TTL Standard				
CMOS 4000B				
comentários				

FAMÍLIA	Velocidade	Fan-Out	Manuseio
TTL Standard			
CMOS 4000B			
comentários			

Tabela 9.18

APÊNDICE

Respostas aos Exercícios Propostos

Capítulo 1 - Sistemas de Numeração

1.6.1
- **a)** 38_{10}
- **b)** 30_{10}
- **c)** 59_{10}
- **d)** 80_{10}
- **e)** 197_{10}
- **f)** 214_{10}
- **g)** 13109_{10}

1.6.2
- **a)** 1001110_2
- **b)** 1100110_2
- **c)** 11010111_2
- **d)** 110010100_2
- **e)** 1100101000_2
- **f)** 1010100110101_2
- **g)** 11111111111111_2

1.6.3
- **a)** 10
- **b)** 4
- **c)** 2
- **d)** 5
- **e)** 6
- **f)** 6
- **g)** 3

1.6.4
- **a)** $3{,}75_{10}$
- **b)** $8{,}0625_{10}$
- **c)** $10{,}625_{10}$
- **d)** $12{,}8125_{10}$
- **e)** $19{,}59375_{10}$
- **f)** $24{,}203125_{10}$
- **g)** $33{,}39065_{10}$

1.6.5 **a)** $0,001_2$
b) $0,0001_2$
c) $0,10110011_2$
d) $0,11101011100001_2$
e) $111,111001100_2$
f) $101111,0111100001_2$
g) $110101,011000110011100111_2$

1.6.6 **a)** 12_{10}
b) 55_{10}
c) 107_{10}
d) 868_{10}
e) 1075_{10}

1.6.7 Porque o algarismo 8 não pertence ao sistema octal.

1.6.8 **a)** 153_8
b) 271_8
c) 4000_8
d) 10001_8
e) 13042_8

1.6.9 **a)** 100111111_2
b) 1101010011_2
c) 100111110100_2
d) 110111100000_2
e) 1000000010001_2

1.6.10 **a)** 13_8
b) 234_8
c) 656_8
d) 1001_8
e) 1505_8

1.6.11 **a)** 1145_{10}
b) 1195_{10}
c) 3038_{10}
d) 61642_{10}
e) 11583_{10}

1.6.12 **a)** $1E6_{16}$
b) $7D0_{16}$
c) 1000_{16}
d) $15B3_{16}$
e) $8A97_{16}$

1.6.13 a) 10000100_2

b) 1111111_2

c) 11101110001100_2

d) 100011111111101_2

e) 1111000111001101_2

1.6.14 722_8 e 4317_8

1.6.15 a) 13_{16}

b) $39C_{16}$

c) 993_{16}

d) $3EF2_{16}$

e) 8022_{16}

1.6.16 $E40_{16}$ e $B33_{16}$

1.6.17 a) 10001_2

b) 101111_2

c) 101010_2

d) 1110110_2

e) 100001100_2

1.6.18 a) 10_2

b) 111_2

c) 1111_2

d) 111110_2

e) 100_2

1.6.19 a) 111111_2

b) 1111101_2

c) 101111010_2

d) 10110100_2

e) 101111100_2

1.6.20 01100001_2 e 11111001_2

1.6.21 -50_{10}

1.6.22 a) 10001011_2

b) 00111101_2

1.6.23 a) 0101_2

b) 011111_2

c) 01000011_2

d) 00101100_2

e) 10101101_2

1.6.24 -73_{10}

1.6.25 a) 110_2

b) 1010011_2

c) -10101111_2

d) 01000111_2

e) -11100010_2

1.6.26 a) 45_8

b) 6_{16}

c) -37_{16}

d) 40_{16}

e) $-3F_{16}$

Capítulo 2 - Funções e Portas Lógicas

2.9.1

(a) NE **(b) NOU**

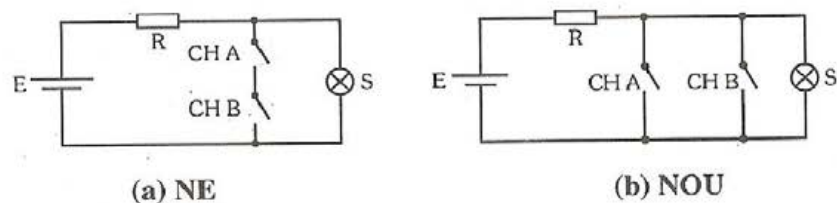



Figura A.1

2.9.2 $S = \left[\overline{(A + B)\left(\overline{AC}\right) + \left(\overline{\overline{B} + D}\right)}\right]$

2.9.3 $S = \left[\overline{\left(\overline{B\overline{D} + A}\right)\left(\overline{B}D + CD\right)}\right]\left[\overline{C + \left(\overline{A} + C\right)\left(\overline{BD}\right)}\right]$

2.9.4 $S = B \oplus D + C\left[\overline{\left(\overline{A\overline{C}D}\right) + \left(\overline{\overline{A} + B + \overline{C}}\right)}\right] + \left[\overline{\left(\overline{\overline{A} + B + \overline{C}}\right)D}\right]$

2.9.5

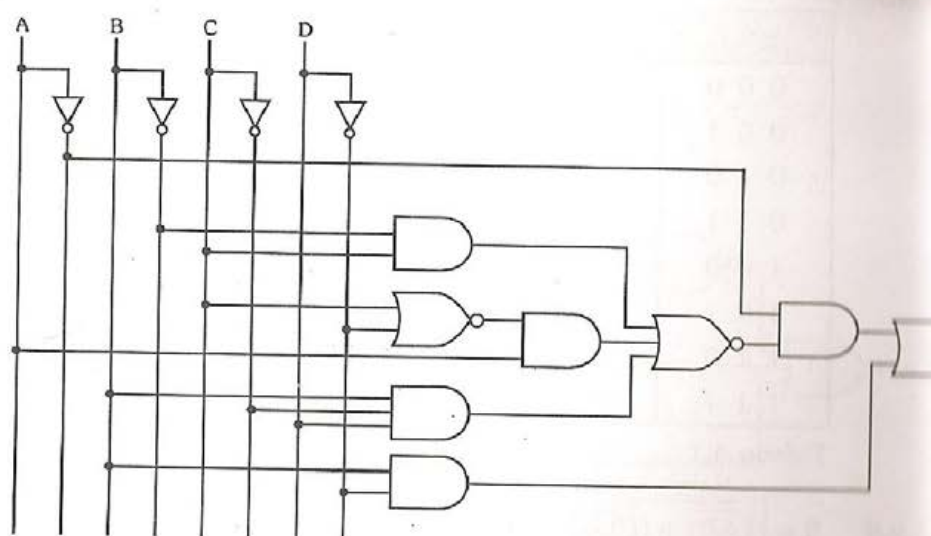



Figura A.2

2.9.6

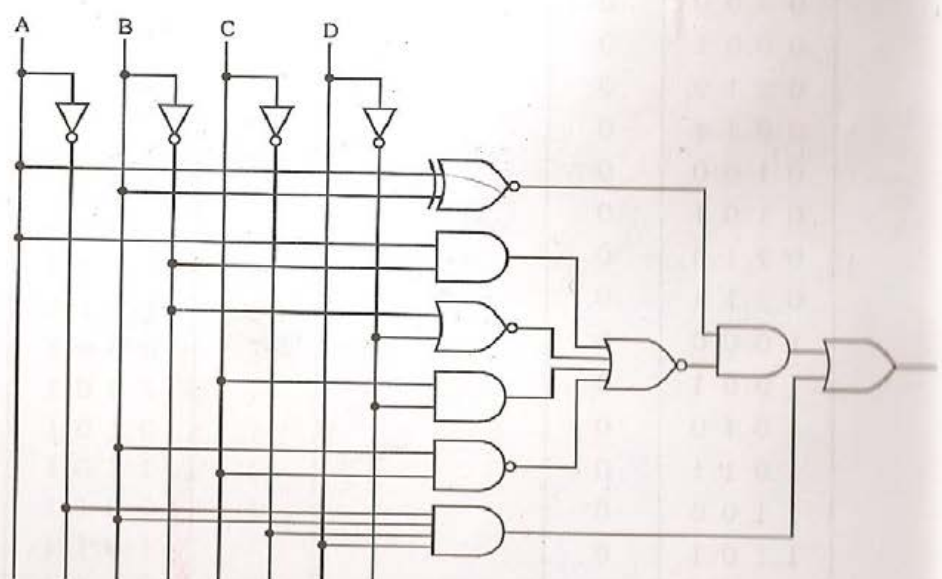



Figura A.3

2.9.7

A B C	S
0 0 0	1
0 0 1	0
0 1 0	0
0 1 1	0
1 0 0	0
1 0 1	0
1 1 0	1
1 1 1	0

Tabela A.1

2.9.8 $S = \left[\overline{(\overline{AB}) + (\overline{C\overline{D}})} \right]$

A B C D	S
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 0 1	0
0 1 1 0	0
0 1 1 1	0
1 0 0 0	0
1 0 0 1	0
1 0 1 0	0
1 0 1 1	0
1 1 0 0	0
1 1 0 1	0
1 1 1 0	1
1 1 1 1	0

Tabela A.2

2.9.9

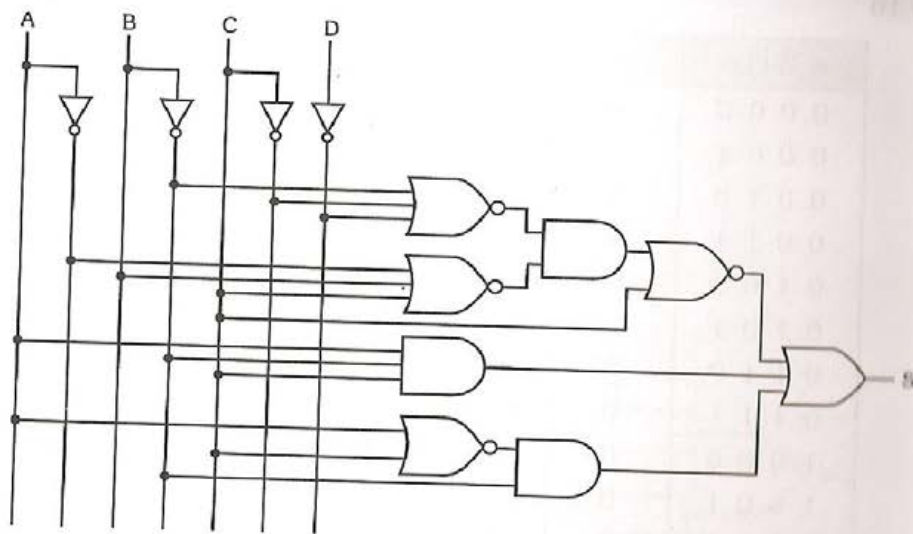



Figura A.4

A B C D	S
0 0 0 0	1
0 0 0 1	1
0 0 1 0	0
0 0 1 1	0
0 1 0 0	1
0 1 0 1	1
0 1 1 0	0
0 1 1 1	0
1 0 0 0	1
1 0 0 1	1
1 0 1 0	1
1 0 1 1	1
1 1 0 0	1
1 1 0 1	1
1 1 1 0	0
1 1 1 1	0

Tabela A.3

2.9.10

A B C D	S
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 0 1	0
0 1 1 0	0
0 1 1 1	0
1 0 0 0	0
1 0 0 1	0
1 0 1 0	0
1 0 1 1	1
1 1 0 0	1
1 1 0 1	0
1 1 1 0	1
1 1 1 1	0

Tabela A.4

2.9.11

A B C	$A \odot (B \oplus C)$	$A \oplus (B \odot C)$
0 0 0	1	1
0 0 1	0	0
0 1 0	0	0
0 1 1	1	1
1 0 0	0	0
1 0 1	1	1
1 1 0	1	1
1 1 1	0	0

Tabela A.5

2.9.12

$S = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}\,B\,C + A\,\overline{B}\,\overline{C} + A\,B\,C$

2.9.13

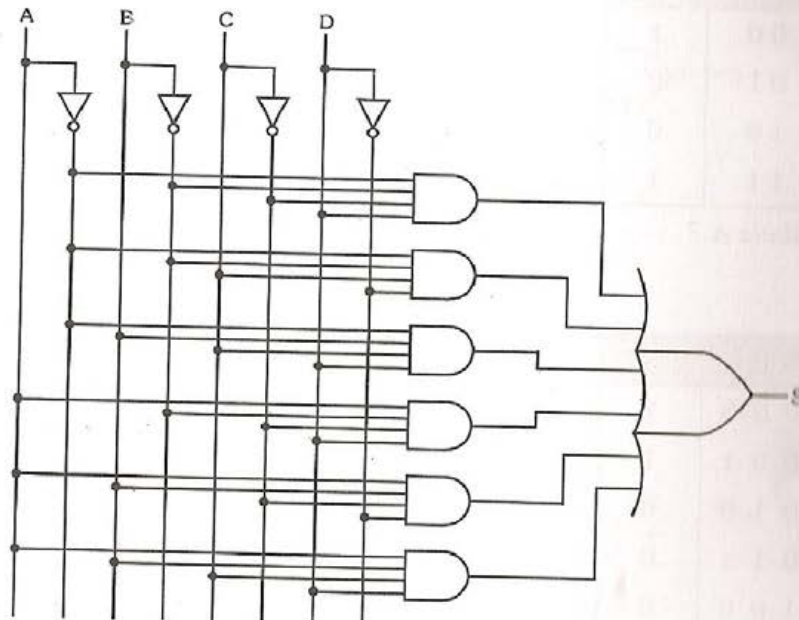



Figura A.5

2.9.14

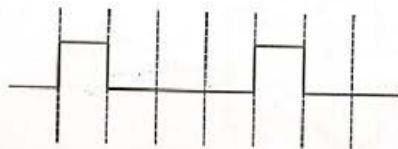



Figura A.6

2.9.15

A B	S
0 0	0
0 1	1
1 0	1
1 1	0

Tabela A.6

2.9.16

A B	S
0 0	1
0 1	0
1 0	0
1 1	1

Tabela A.7

2.9.17

A B C	S
0 0 0	1
0 0 1	1
0 1 0	0
0 1 1	0
1 0 0	0
1 0 1	0
1 1 0	1
1 1 1	0

Tabela A.8

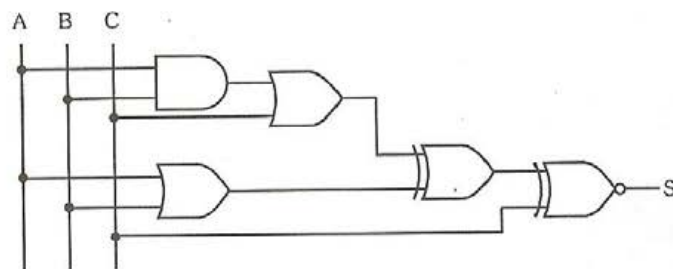



Figura A.7

5.6.4

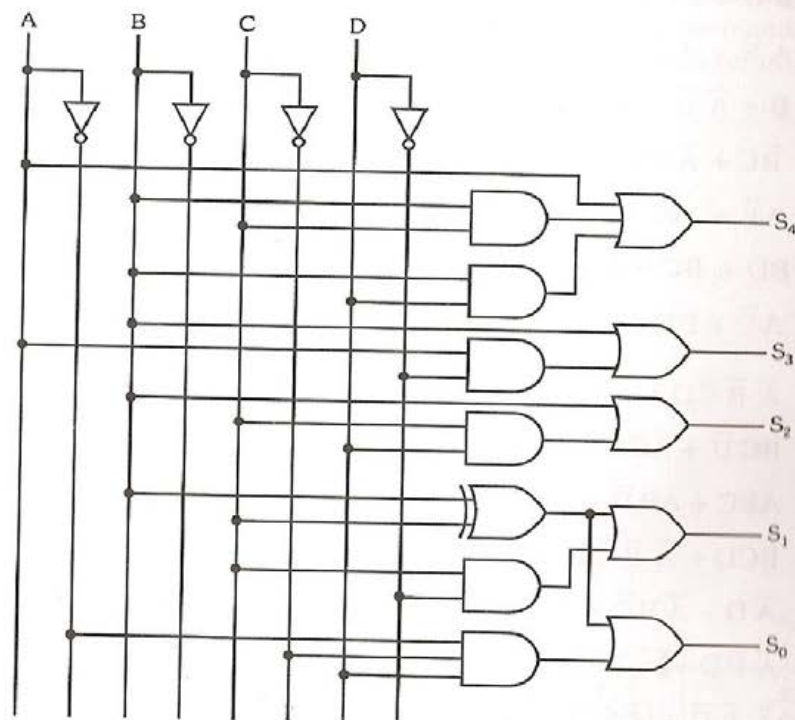



Figura A.26

5.6.5 $S_A = B\overline{C} + BD$

$S_B = \overline{B}D + C\overline{D} + AD$

$S_C = \overline{D}$

$S_D = A\overline{D} + BC\overline{D} + \overline{B}CD + \overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}D$

5.6.6 $a = B + AC$

$b = \overline{A} + \overline{C}$

$c = \overline{B} + C$

$d = B + AC$

$e = B\overline{C}$

$f = A$

$g = A + B$

5.6.7 $a = C$

$b = \overline{A}\,\overline{B}\,\overline{C} + BC + AB$

$c = \overline{A}\,\overline{C} + B$

$d = \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$

$e = A\overline{B}C$

$f = \overline{A}\,\overline{B}C + ABC$

$g = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}BC$

2.9.21

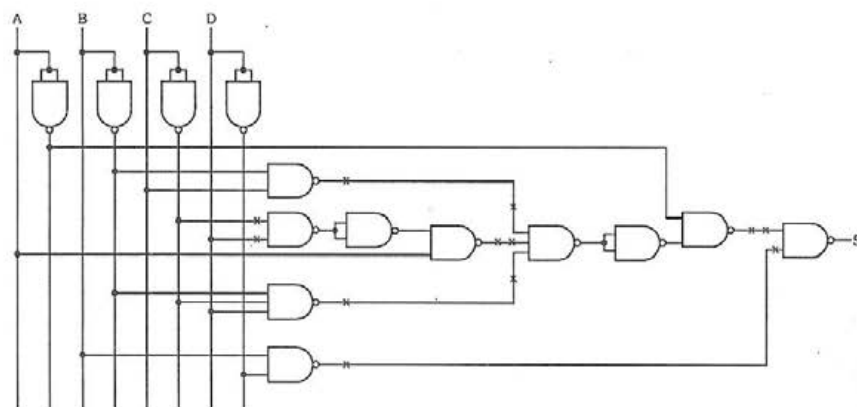



Figura A.11

2.9.22

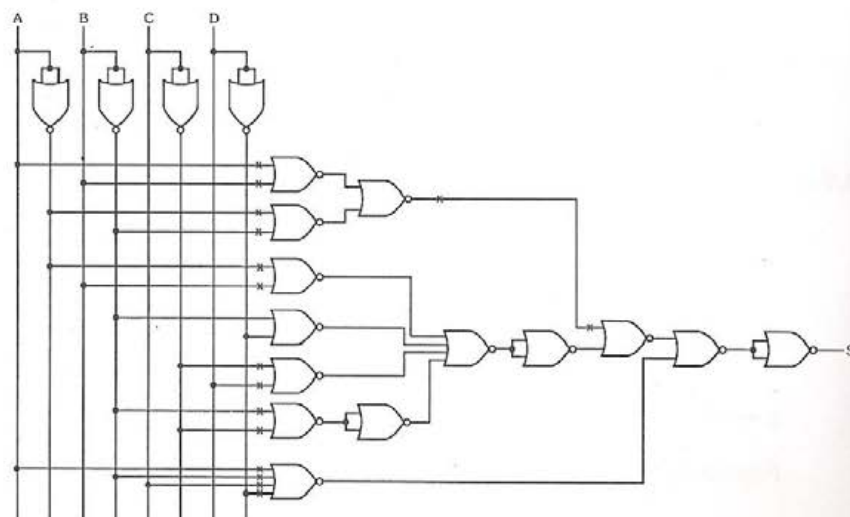



Figura A.12

2.9.23

A B C D	S
0 0 0 0	0
0 0 0 1	0
0 0 1 0	0
0 0 1 1	0
0 1 0 0	0
0 1 0 1	1
0 1 1 0	0
0 1 1 1	0
1 0 0 0	0
1 0 0 1	0
1 0 1 0	0
1 0 1 1	0
1 1 0 0	0
1 1 0 1	0
1 1 1 0	0
1 1 1 1	0

Tabela A.9

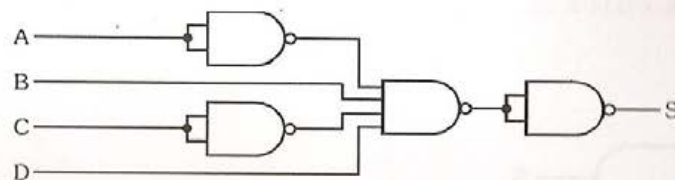



Figura A.13

2.9.24

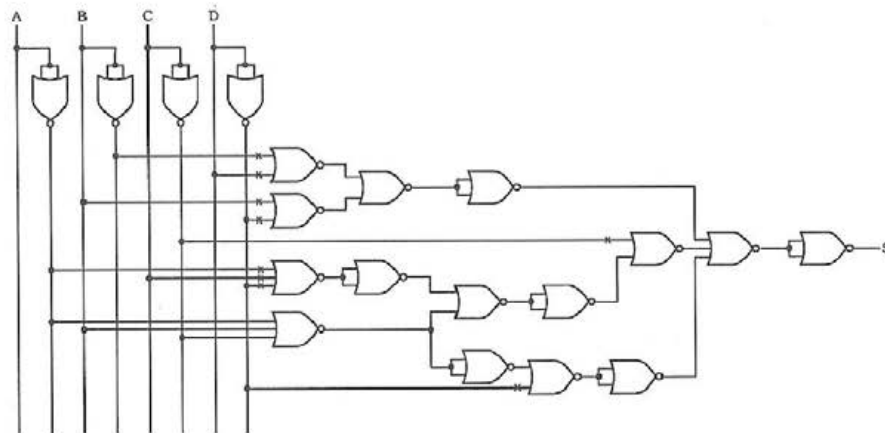



Figura A.14

Capítulo 3 - Álgebra de Boole e Simplificação de Circuitos Lógicos

3.10.1 a) $S = \overline{A}C + B$ b) $S = AB + C\overline{D}$

3.10.2 $S = \overline{C} + \overline{A}\,\overline{B}$

3.10.3 $S = C\overline{D} + AB + AD + AC$

3.10.4 $S = \overline{A}\,\overline{D} + \overline{A}\,\overline{B}\,\overline{C} + ABD$

3.10.5 $S = B + AC$

3.10.6

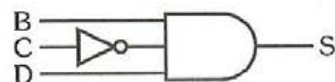



Figura A.15

3.10.7 $S = A\overline{C}\,\overline{D}$

3.10.8 $S = A \odot (B \oplus C)$

$$S = \overline{A}\left(\overline{B \oplus C}\right) + A\left(B \oplus C\right)$$

$$S = \overline{A}\,(B \odot C) + A\,(\overline{B \odot C})$$

$$S = A \oplus (B \odot C)$$

3.10.9 $S_1 = A + \overline{B}$ $S_2 = \overline{A}$

3.10.10 $S_1 = \overline{B}\,\overline{C} + AC + \overline{A}B$ $S_3 = \overline{B}C + A\overline{C}$

$S_2 = \overline{B} + \overline{C}$ $S_4 = \overline{A}\,\overline{B}C + A\overline{C} + AB + B\overline{C}$

3.10.11 $S_1 = \overline{B} + \overline{C}\,\overline{D} + CD$ $S_3 = \overline{A}B\overline{D} + B\overline{C}D + \overline{B}C\overline{D}$

$S_2 = \overline{A}\,\overline{D} + BD + A\overline{B}\,\overline{C}$ $S_4 = \overline{A}B\overline{C} + \overline{A}CD + ABC + A\overline{C}$ ...

3.10.12 **a)** $S = A\overline{B} + AC + \overline{A}B$ **c)** $S = \overline{A} + \overline{B}$

b) $S = \overline{B}\,\overline{C}\,\overline{D} + A\overline{C}D + BCD + \overline{A}\,\overline{B}C$ **d)** $S = \overline{B}C + AC + BD + B\overline{C}$

3.10.13 $S_1 = CE + \overline{A}\,\overline{B}\,\overline{C} + \overline{B}\,\overline{C}\,\overline{D} + \overline{A}BC\overline{D} + B\overline{C}D\overline{E}$

$S_2 = C + \overline{E} + A\overline{B}D$

3.10.14 $S_1 = A + B$ $S_2 = \overline{A}\,\overline{B} + AC$

3.10.15 $S_1 = \overline{B} + \overline{D}$ $S_3 = B\overline{D} + A\overline{B}\,\overline{C} + ABC + \overline{A}\,\overline{B}C$ ...

$S_2 = BD + AC + \overline{B}\,\overline{D}$ $S_4 = B\overline{C} + AD + CD + AB$

3.10.16

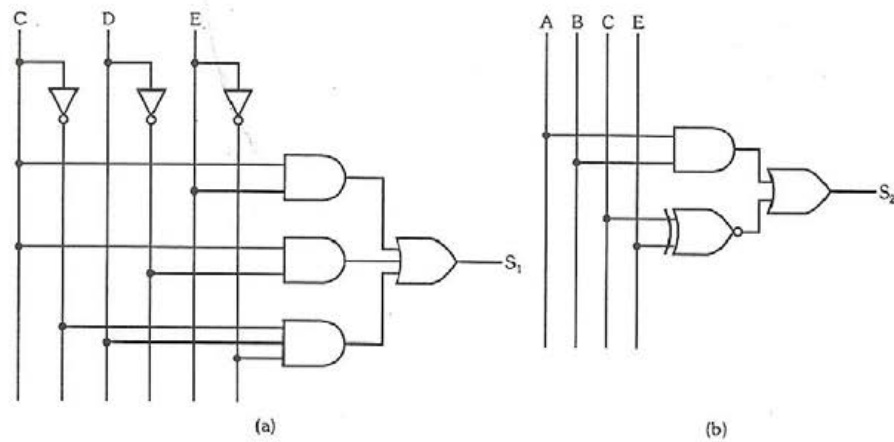



Figura A.16

3.10.17 $S = \overline{A}B + B\overline{D}$

3.10.18

A B C D	$\overline{A \oplus B \oplus C \oplus D}$	$A \odot B \odot C \odot D$
0 0 0 0	1	1
0 0 0 1	0	0
0 0 1 0	0	0
0 0 1 1	1	1
0 1 0 0	0	0
0 1 0 1	1	1
0 1 1 0	1	1
0 1 1 1	0	0
1 0 0 0	0	0
1 0 0 1	1	1
1 0 1 0	1	1
1 0 1 1	0	0
1 1 0 0	1	1
1 1 0 1	0	0
1 1 1 0	0	0
1 1 1 1	1	1

Tabela A.10

4.3.1

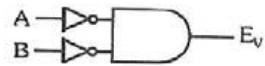



Figura A.17

4.3.2 $V_1 = AB + \overline{B}\,\overline{C}$ $\qquad Vm_1 = \overline{A}B + \overline{B}C$

$V_2 = \overline{A}B$ $\qquad Vm_2 = A + \overline{B}$

$V_3 = \overline{B}C$ $\qquad Vm_3 = B + \overline{C}$

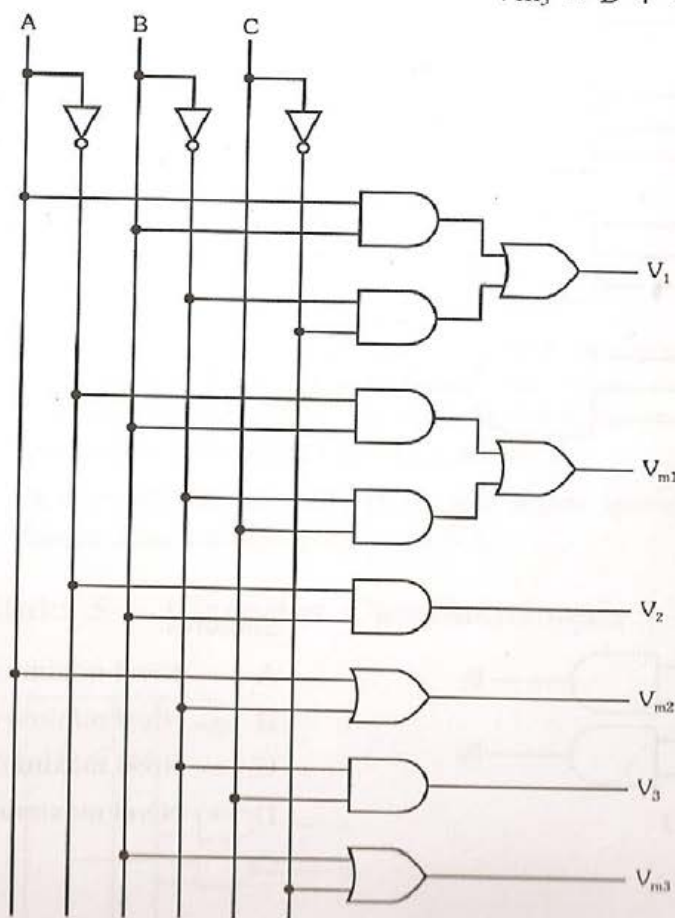



Figura A.18

4.3.3

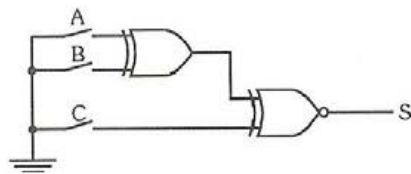



Figura A.19

4.3.4

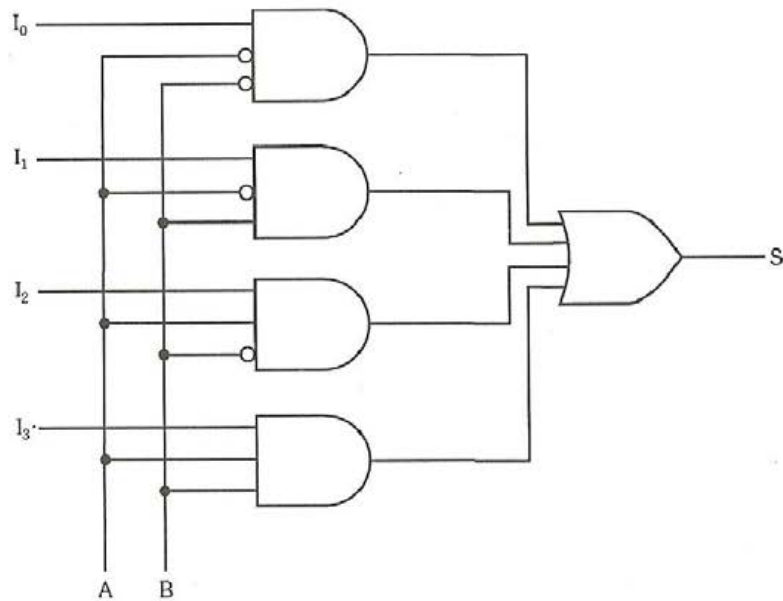



Figura A.20

4.3.5

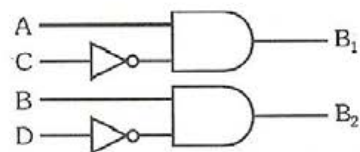



Figura A.21

Sensores:

A → Nível mínimo de T_1

B → Nível mínimo de T_2

C → Nível máximo de T_2

D → Nível máximo de T_3

4.3.6 A bomba B_1 será ligada quando houver líquido em A (sensor T_1) e nã houver em C (sensor de nível máximo de T_2), pois $B_1 = A\overline{C}$. A bomb B_2 será ligada quando houver líquido em B (sensor T_2) e não houve em D (sensor de nível máximo de T_3), pois $B_2 = B\overline{D}$.

4.3.7 I_1 → botão de cima

I_2 → botão de baixo

A → sensor de nível máximo

B → sensor de nível colocado na metade do tanque

E_{VE} → eletroválvula de entrada

E_{VS} → eletroválvula de saída

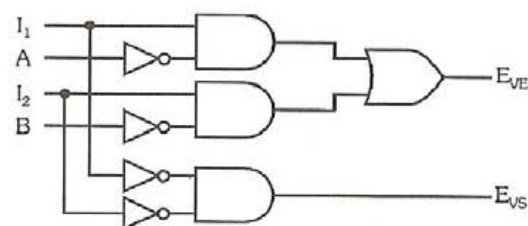



Figura A.22

4.3.8 A eletroválvula de entrada (E_{VE}) será ligada quando o botão de cima for acionado e não houver líquido no sensor A (nível máximo) ou quando o botão de baixo for acionado e não houver líquido no sensor B (meio do tanque), pois $E_{VE} = I_1\overline{A} + I_2\overline{B}$.

A eletroválvula de saída (E_{VS}) será ligada quando ambos os botões forem desacionados, pois $E_{VS} = \overline{I_1}\,\overline{I_2}$.

Capítulo 5 - Circuitos Combinacionais - 2ª Parte

5.6.1

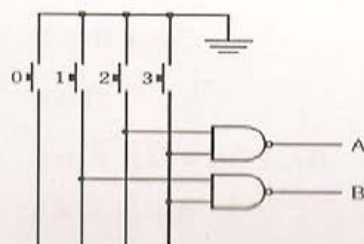



Figura A.23

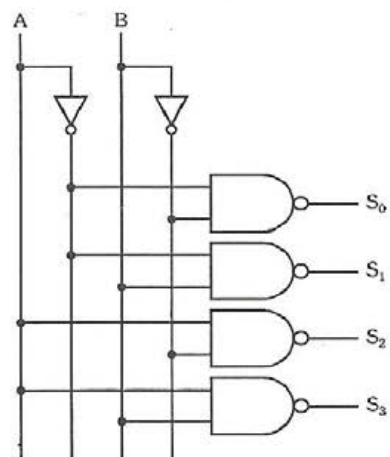
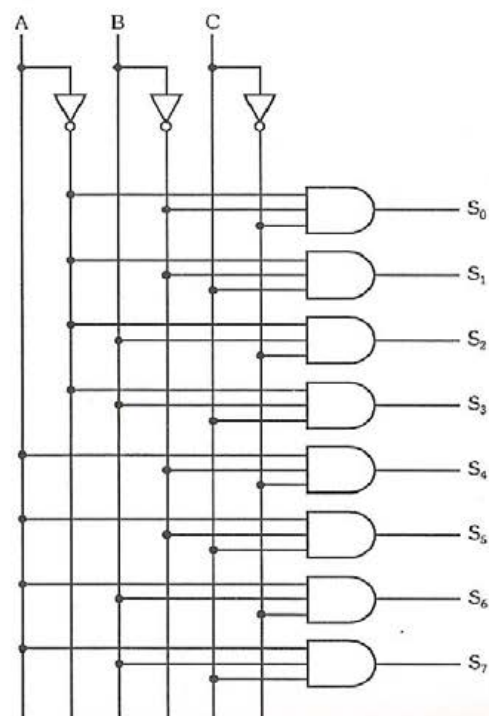
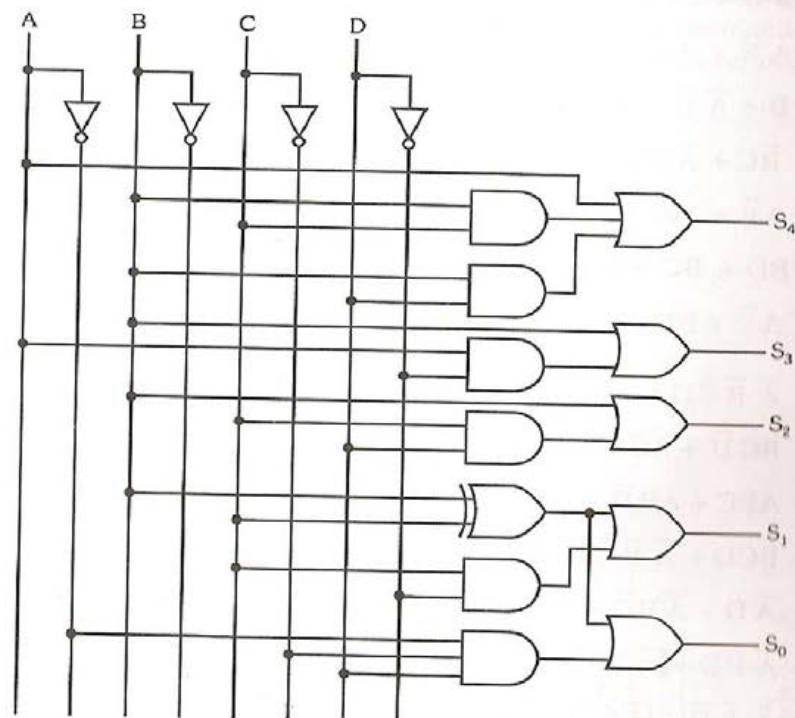
5.6.2
A
B
S0
S1
S2
S3
Figura A.24
5.6.3
A
B
C
S0
S1
S2
S3
S4
S5
S6
S7
Figura A.25
492
Elementos de Eletrônica Digital

5.6.4

Figura A.26

5.6.5 $S_A = B\overline{C} + BD$

$S_B = \overline{B}D + C\overline{D} + AD$

$S_C = \overline{D}$

$S_D = A\overline{D} + BC\overline{D} + \overline{B}CD + \overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}D$

5.6.6 $a = B + AC$

$b = \overline{A} + \overline{C}$

$c = \overline{B} + C$

$d = B + AC$

$e = B\overline{C}$

$f = A$

$g = A + B$

5.6.7 $a = C$

$b = \overline{A}\,\overline{B}\,\overline{C} + BC + AB$

$c = \overline{A}\,\overline{C} + B$

$d = \overline{A}B\overline{C} + A\overline{B}\,\overline{C} + ABC$

$e = A\overline{B}C$

$f = \overline{A}\,\overline{B}C + ABC$

$g = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}BC$

5.6.8 $a = \overline{B}\,\overline{D} + BD + B\overline{C} + A\overline{C} + \overline{A}\,\overline{B}C$

$b = \overline{A}\,\overline{B} + \overline{A}\,\overline{D} + AB\overline{C} + ACD$

$c = B + \overline{A}\,\overline{D} + \overline{A}\,\overline{C} + ACD$

$d = \overline{B}C + \overline{A}\,\overline{B}\,\overline{D} + \overline{A}BD + AB\overline{D} + A\overline{C}D$

$e = A\overline{B} + AC + A\overline{D} + \overline{B}\,\overline{C}\,\overline{D} + \overline{A}B\overline{C}D + \overline{B}CD$

$f = BD + BC + A\overline{C} + A\overline{D} + \overline{B}\,\overline{C}\,\overline{D}$

$g = A\overline{C} + BD + \overline{A}C + AD + BC$

5.6.9 $a = \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}B\overline{C}\,\overline{D} + AB\overline{C}D + A\overline{B}CD$

$b = BC\overline{D} + ACD + AB\overline{D} + \overline{A}B\overline{C}D$

$c = ABC + AB\overline{D} + \overline{A}\,\overline{B}C\overline{D}$

$d = BCD + \overline{A}\,\overline{B}\,\overline{C}D + \overline{A}B\overline{C}\,\overline{D} + A\overline{B}C\overline{D}$

$e = \overline{A}D + \overline{A}B\overline{C} + \overline{B}\,\overline{C}D$

$f = \overline{A}\,\overline{B}D + \overline{A}\,\overline{B}C + \overline{A}CD + AB\overline{C}D$

$g = \overline{A}\,\overline{B}\,\overline{C} + \overline{A}BCD + AB\overline{C}\,\overline{D}$

5.6.10 Aplicando o 3º (terceiro) número ao terminal T_E.

5.6.11

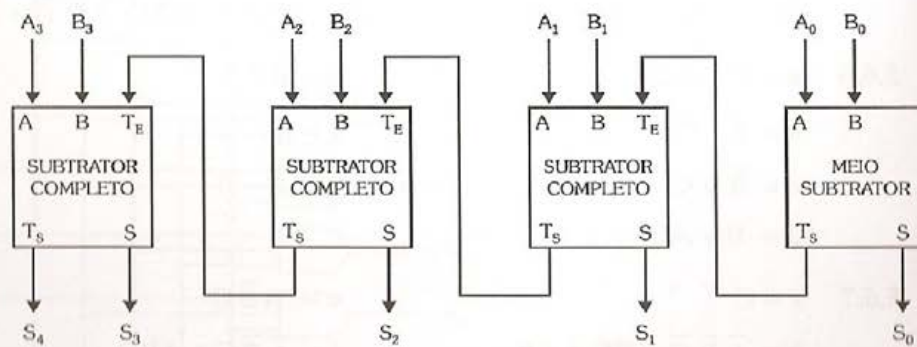



Figura A.27

5.6.12 O sistema apresentará nas saídas S_3, S_2, S_1 e S_0, o resultado da operação na notação do complemento de 2 (negativo), apresentando, ainda, na saída S_4 (T_S do último bloco), um nível 1 que poderá ser utilizado como sinalizador da operação.

5.6.13

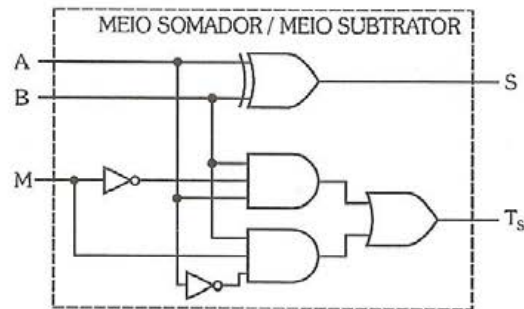



Figura A. 28

5.6.14

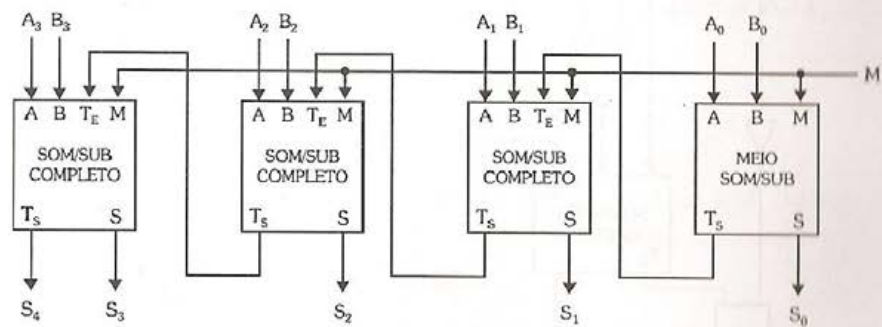



Figura A.29

5.6.15

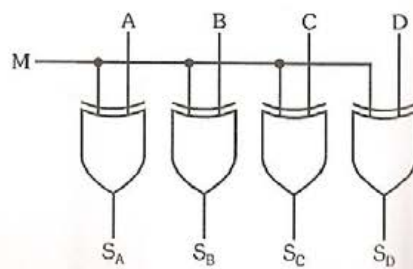



Figura A.30

5.6.16

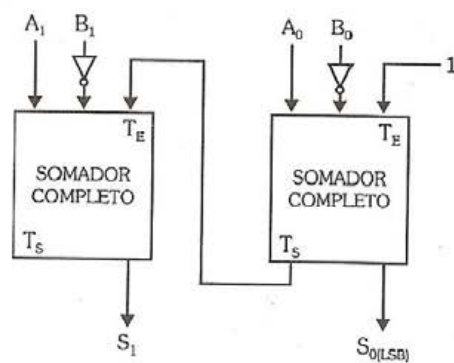



Figura A.31

5.6.17

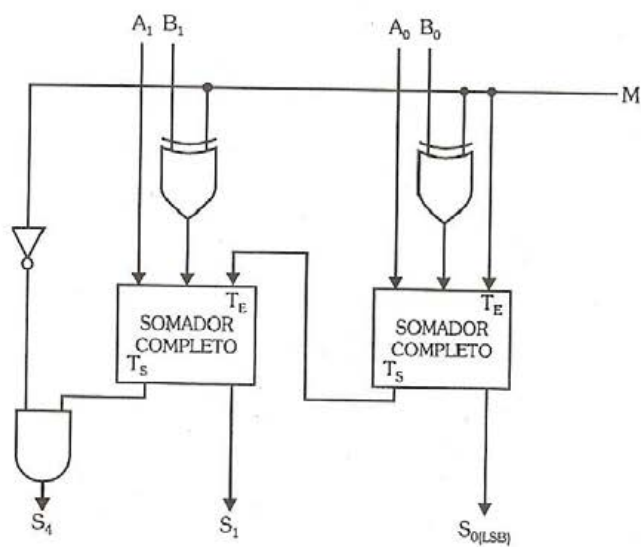



Figura A.32

Capítulo 6 - Flip-Flop, Registradores e Contadores

6.5.1

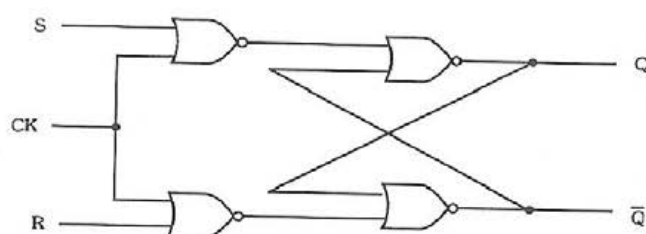



Figura A.33

S R	Qf
0 0	X
0 1	0
1 0	1
1 1	Qa

(a)

CK	Qf
0	RS básic
1	Qa

(b)

Tabela A.11

6.5.2

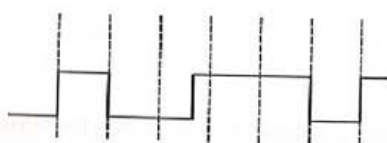



Figura A.34

6.5.3

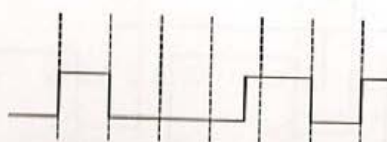



Figura A.35

6.5.4

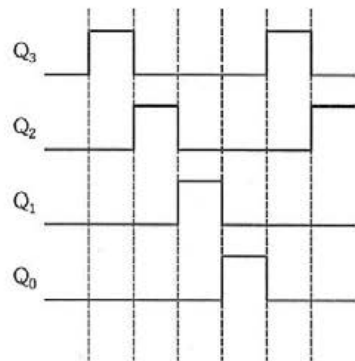



Figura A.36

6.5.5 $Q_3 = 0$, $Q_2 = 0$, $Q_1 = 0$ e $Q_0 = 1$

6.5.6 O nível 1 de Q_0 iria novamente ser aplicado à entrada ES e com as sucessivas descidas de pulso passaria de saída em saída, permanecendo em loop no sistema.

6.5.7

Q_5	Q_4	Q_3	Q_2	Q_1	Q_0
0	1	1	0	0	0

6.5.8 Passou de 6_{10} para 24_{10}, pois foi multiplicado por 4 (2 deslocamentos à esquerda).

6.5.9

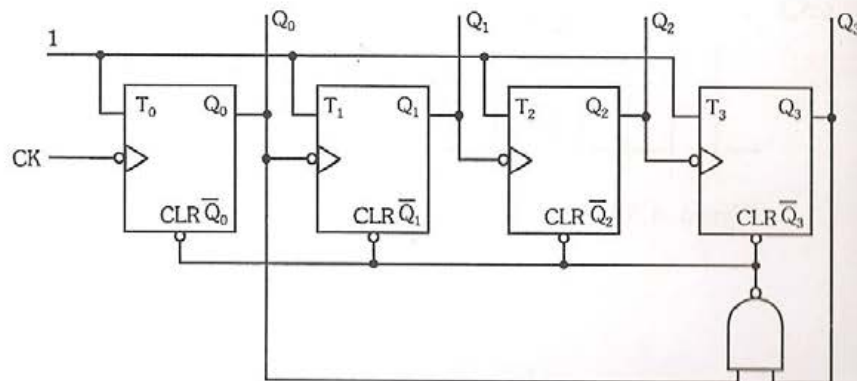



Figura A. 37

6.5.10

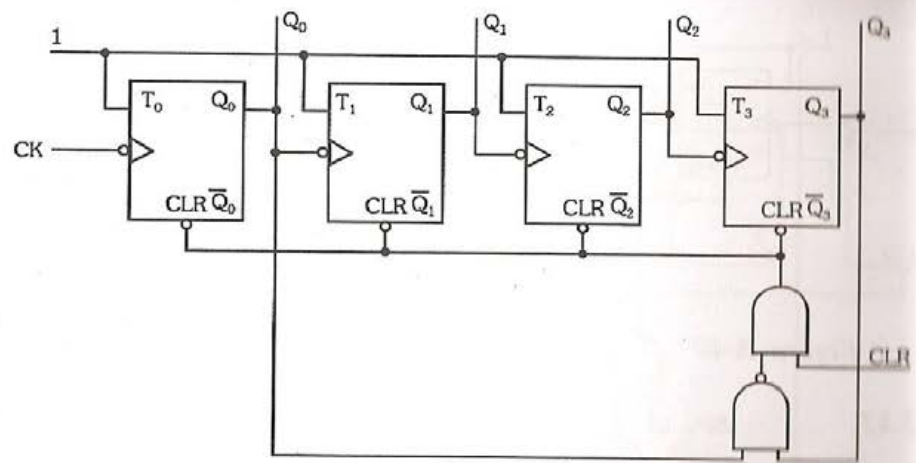



Figura A.38

6.5.11

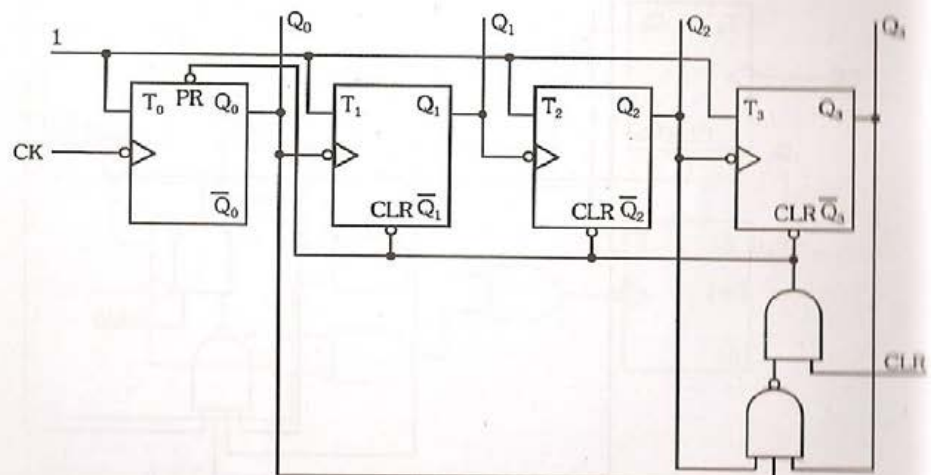



Figura A.39

6.5.12

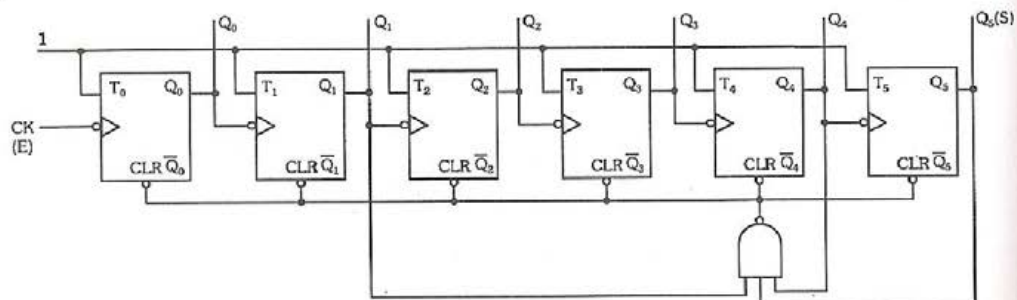



Figura A.40

6.5.13

Figura A.41

6.5.14

Figura A.42

6.5.15

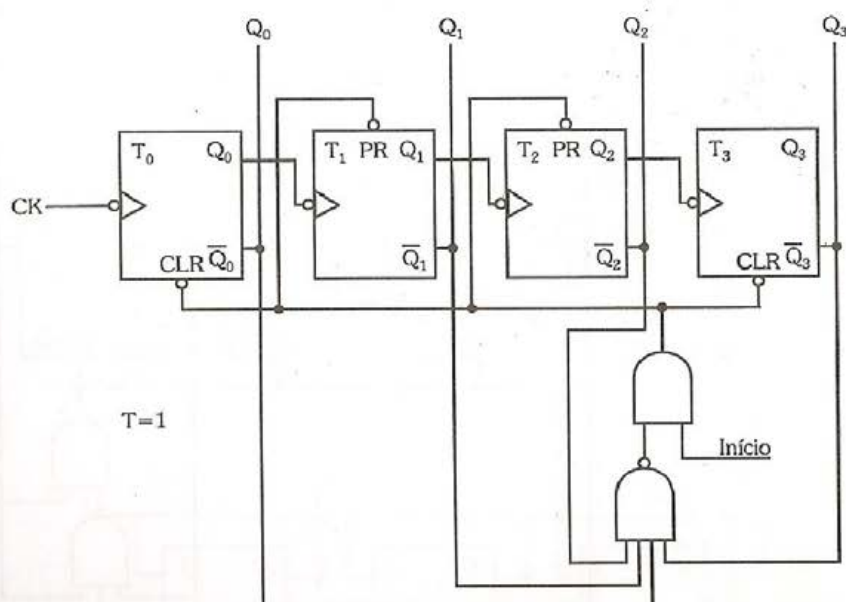



Figura A.43

6.5.16

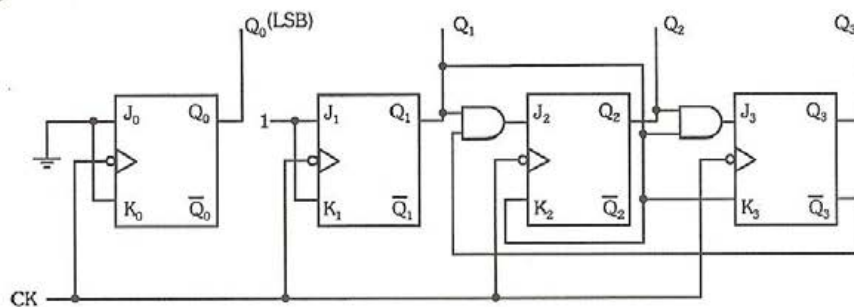



Figura A.44

6.5.17

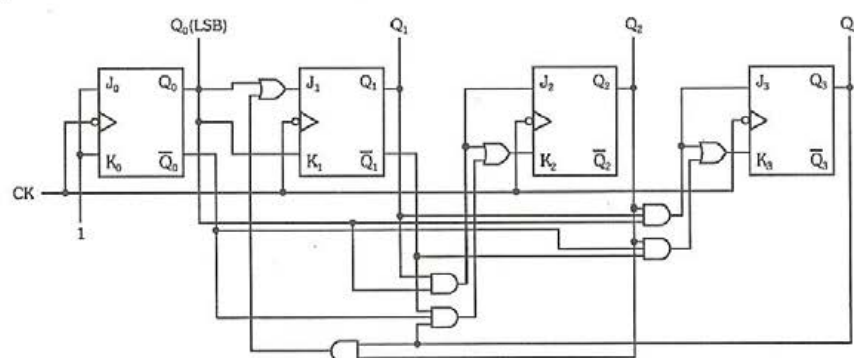



Figura A.45

6.5.18

$J_3 = \overline{Q_2}\,\overline{Q_1}\,\overline{Q_0}$ $K_3 = \overline{Q_0}$

$J_2 = Q_3\overline{Q_0}$ $K_2 = \overline{Q_1}\,\overline{Q_0}$

$J_1 = Q_3\overline{Q_0} + Q_2\overline{Q_0}$ $K_1 = \overline{Q_0}$

$J_0 = 1$ $K_0 = 1$

6.5.19

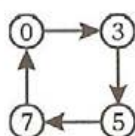



Figura A.46

6.5.20

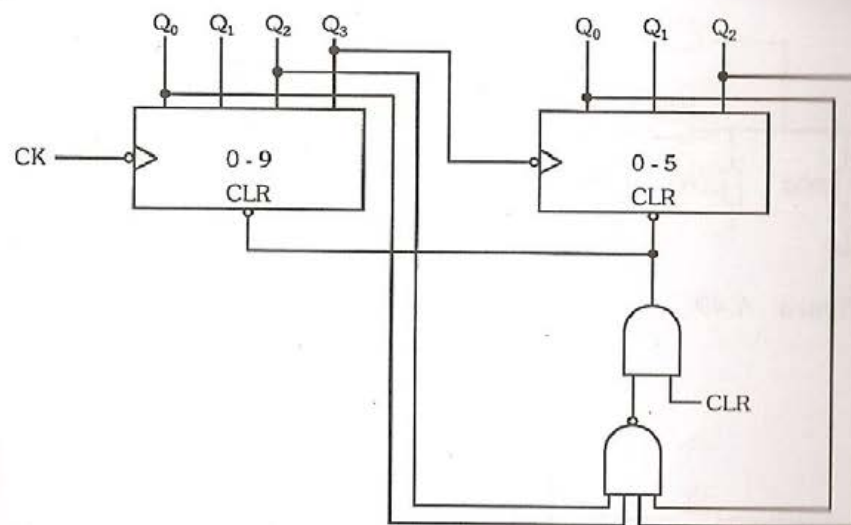



Figura A.47

Capítulo 7 - Conversores Digital-Analógicos e Análogo--Digitais

7.5.1

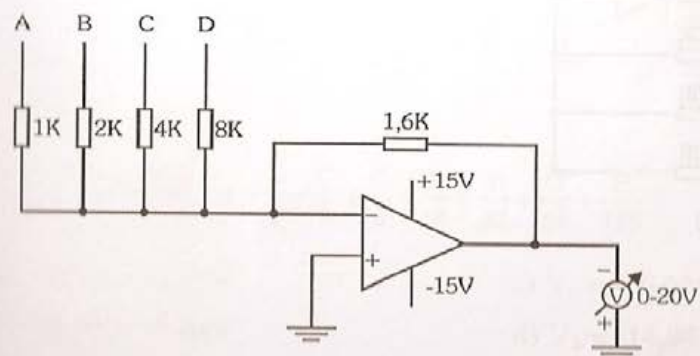



Figura A.48

7.5.2

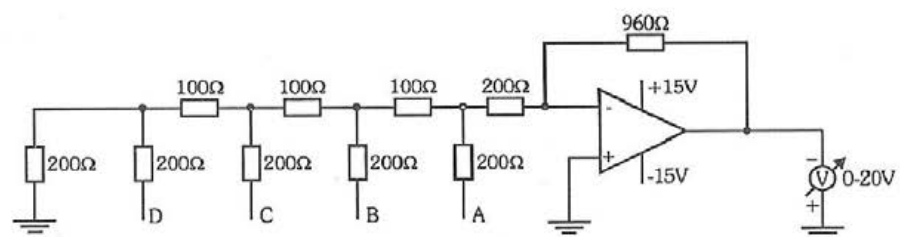



Figura A.49

7.5.3

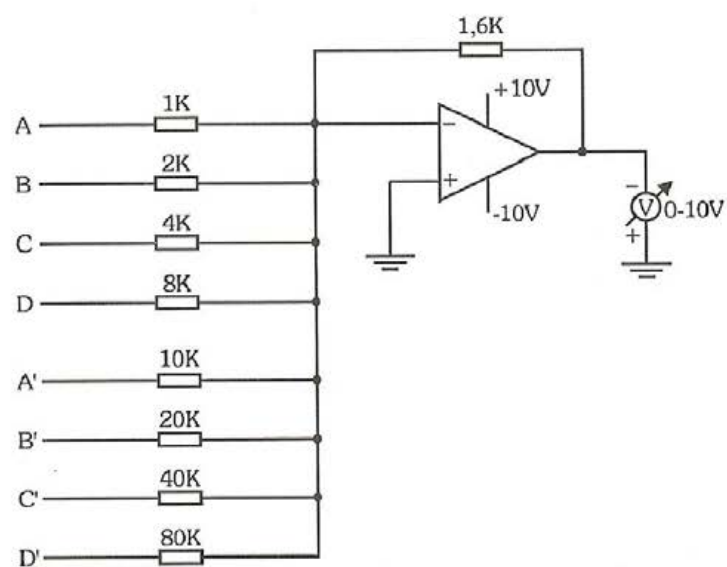



Figura A.50

7.5.4

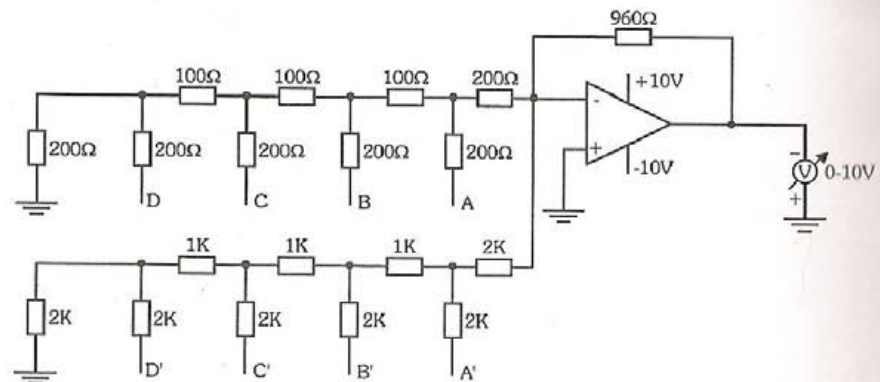



Figura A.51

7.5.5

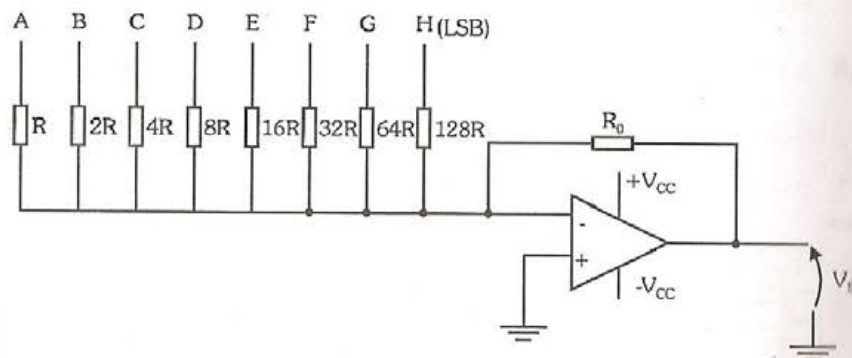



Figura A.52

$$V_S = \frac{-V.R_0}{R}\left(A+\frac{B}{2}+\frac{C}{4}+\frac{D}{8}+\frac{E}{16}+\frac{F}{32}+\frac{G}{64}+\frac{H}{128}\right)$$

7.5.6 **a)** $V_S = -1{,}81V$ **c)** $V_S = -10{,}02V$

b) $V_S = -5{,}04V$ **d)** $V_S = -14{,}94V$

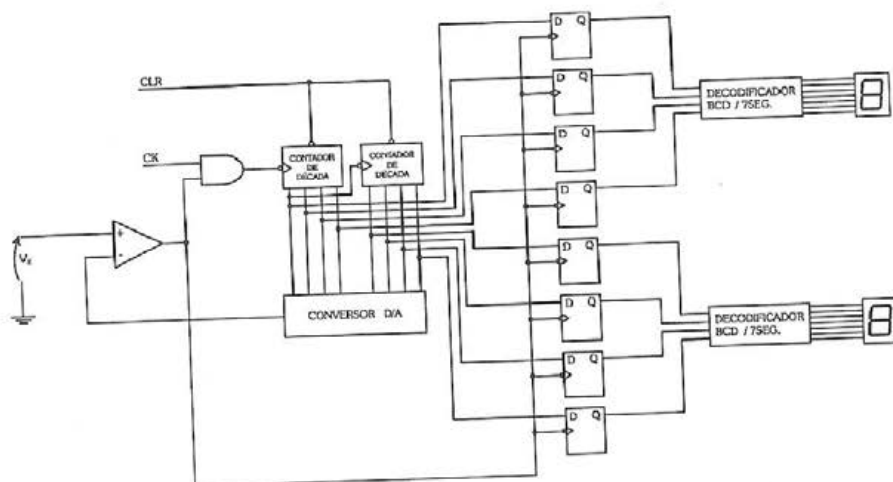
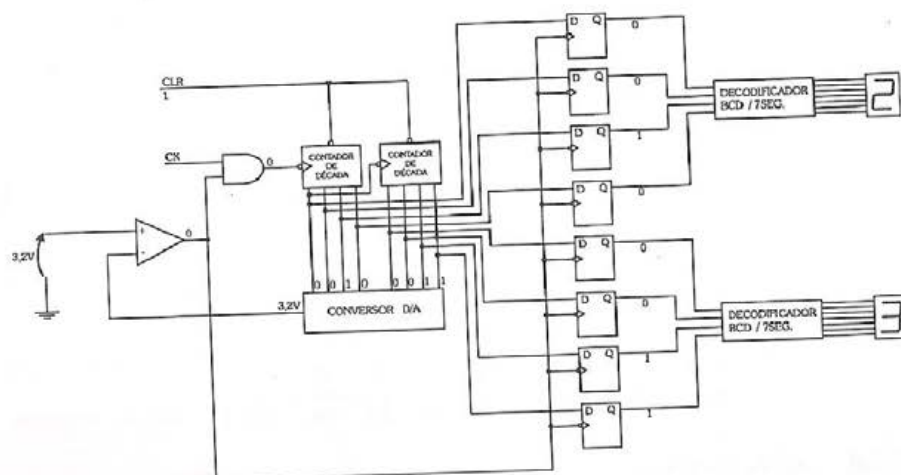



Figura A.53

Figura A.54

7.5.9

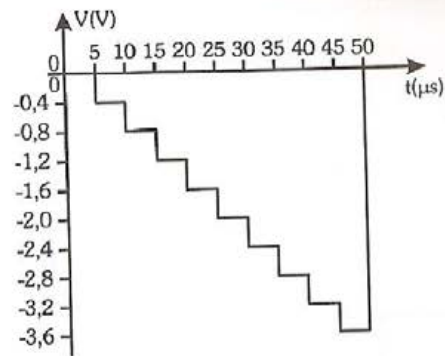



Figura A.55

7.5.10 fclock = 1600 Hz

7.5.11

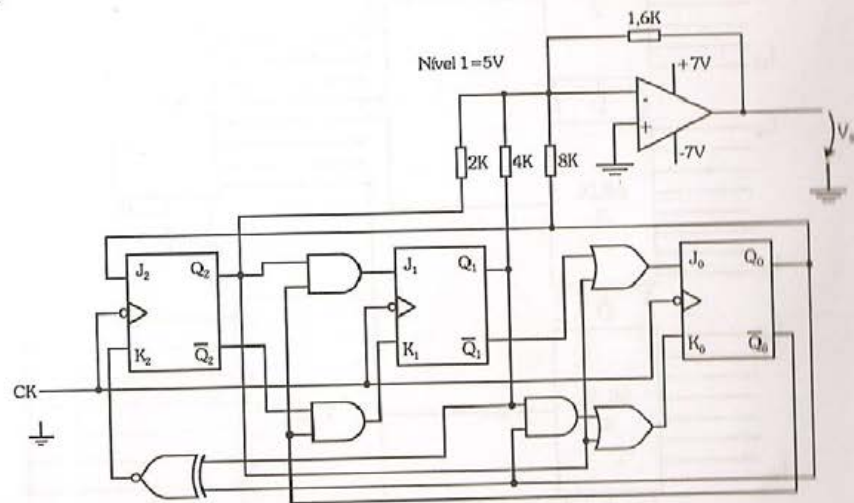



Figura A.56

fclock = 1MHz

7.5.12

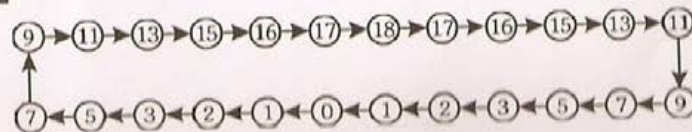



Figura A.57

8.7.1 44 portas E.

8.7.2 40 portas E.

8.7.3

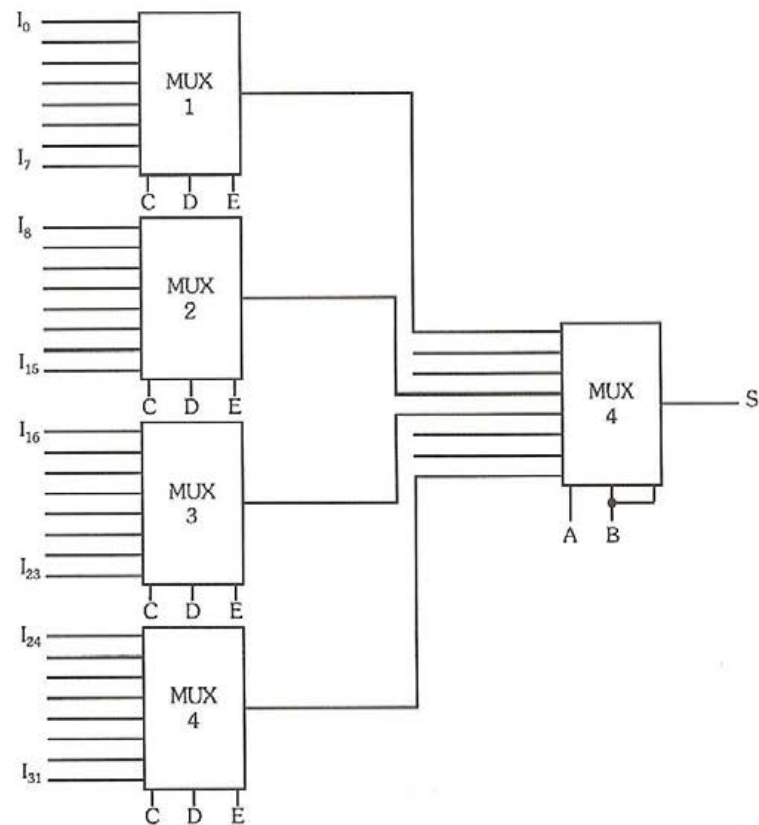



Figura A.58

8.7.4
Figura A.59
8.7.5
+Vcc
I0
I1
I2
I3
I4
I5
I6
I7
MUX
S
A
B
C
Figura A.60
8.7.6
+Vcc
I0
I1
I2
I3
I4
I5
I6
I7
MUX
S
A
B
C
Figura A.61
Apêndice - Respostas aos Exercícios Propostos

P0
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
I0
I1
I2
I3
I4
I5
I6
I7
I8
I9
I10
I11
I12
I13
I14
I15
E
A B C D

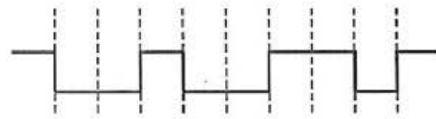
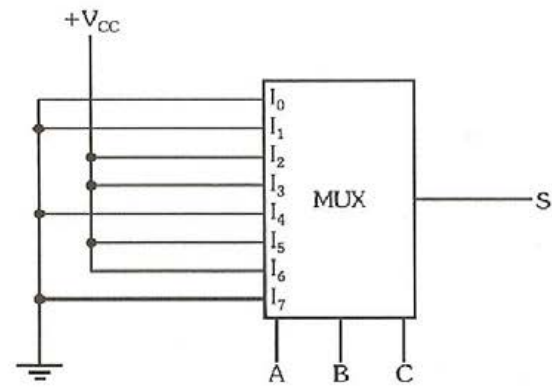
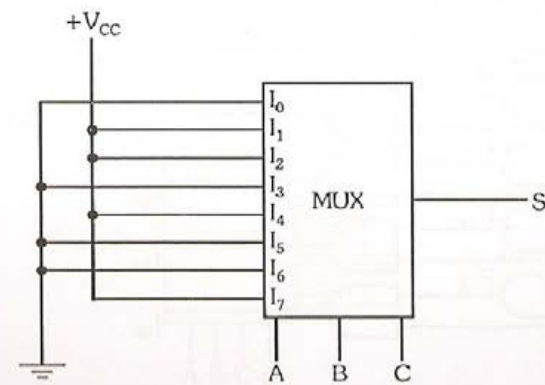
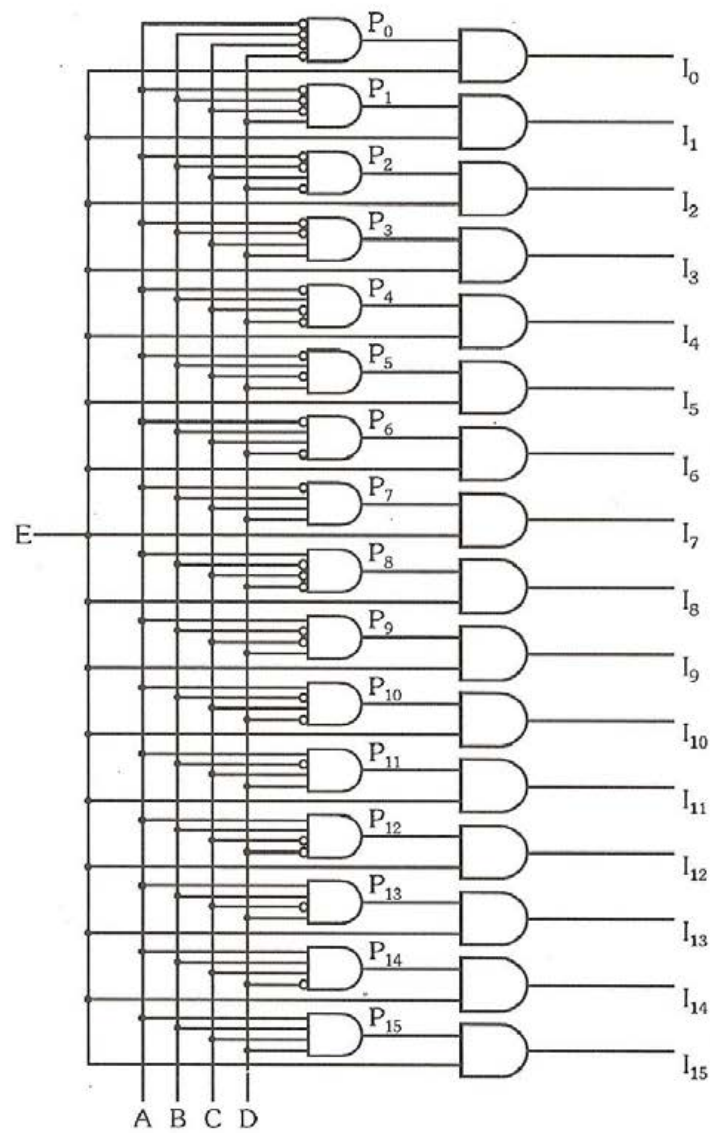



Figura A.62

8.7.8

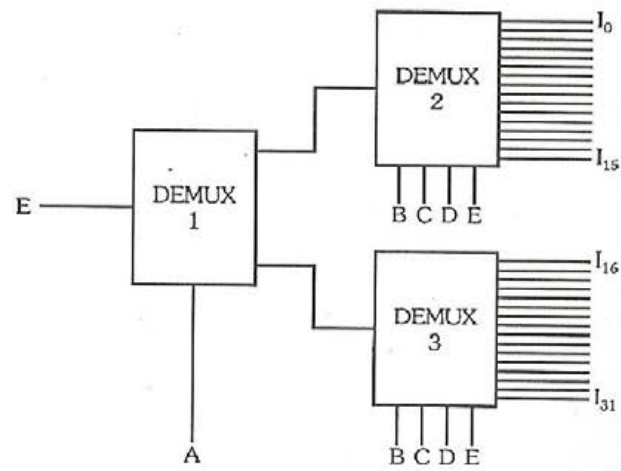



Figura A.63

8.7.9

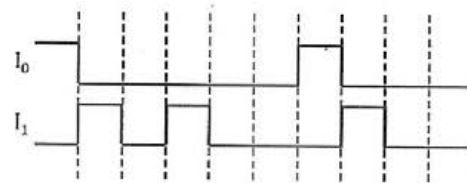



Figura A.64

8.7.10

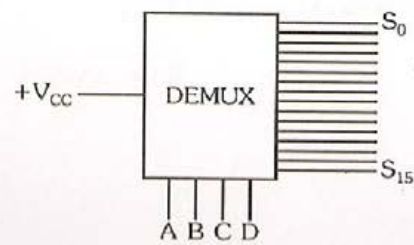



Figura A.65

8.7.11

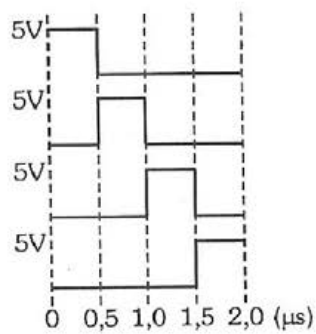



Figura A.66

8.7.12

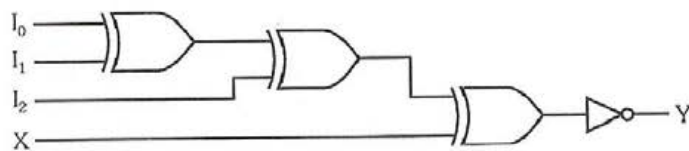



Figura A.67

8.7.13 **a)** Capacidade 2048 bits, endereço: 000_{16} a $1FF_{16}$

b) Capacidade 32768 bits, endereço: 000_{16} a FFF_{16}

c) Capacidade 1M bits, endereço: 00000_{16} a $1FFFF_{16}$

d) Capacidade 32M bits, endereço: 000000_{16} a $1FFFFF_{16}$

8.7.14

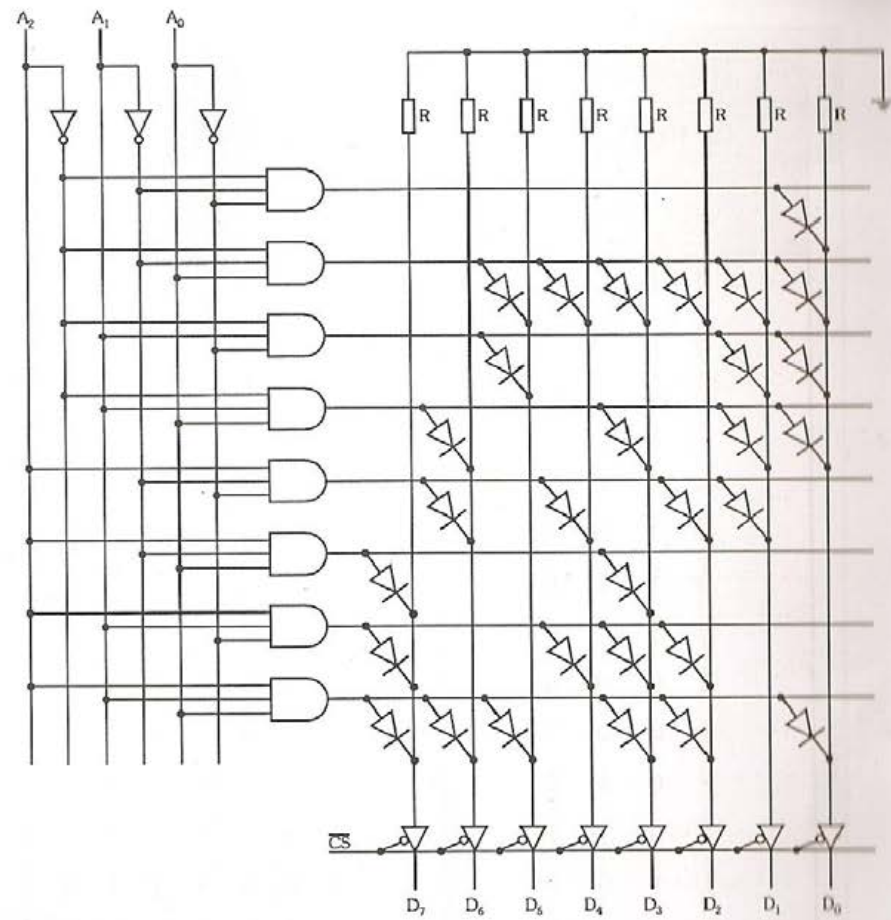



Figura A.68

8.7.15

ENDEREÇAMENTO	DADOS
BINÁRIO	HEXADECIMAL
A_3 A_2 A_1 A_0	D_6 D_5 D_4 D_3 D_2 D_1 D_0 a b c d e f g
0 0 0 0	1 1 1 1 1 1 0
0 0 0 1	0 1 1 0 0 0 0
0 0 1 0	1 1 0 1 1 0 1
0 0 1 1	1 1 1 1 0 0 1
0 1 0 0	0 1 1 0 0 1 1
0 1 0 1	1 0 1 1 0 1 1
0 1 1 0	1 0 1 1 1 1 1
0 1 1 1	1 1 1 0 0 0 0
1 0 0 0	1 1 1 1 1 1 1
1 0 0 1	1 1 1 1 0 1 1
1 0 1 0	1 1 1 0 1 1 1
1 0 1 1	0 0 1 1 1 1 1
1 1 0 0	1 0 0 1 1 1 0
1 1 0 1	0 1 1 1 1 0 1
1 1 1 0	1 0 0 1 1 1 1
1 1 1 1	1 0 0 0 1 1 1

Tabela A.12

Memória 16x7 (112 bits)

8.7.16

ENDEREÇAMENTO					DADOS (Valores digitalizados)			
A_4	A_3	A_2	A_1	A_0	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1	0
0	0	0	1	1	0	0	1	1
0	0	1	0	0	0	1	0	0
0	0	1	0	1	0	1	0	1
0	0	1	1	0	0	1	1	0
0	0	1	1	1	0	1	1	1
0	1	0	0	0	1	0	0	0
0	1	0	0	1	1	0	0	1
0	1	0	1	0	1	0	1	0
0	1	0	1	1	1	0	1	1
0	1	1	0	0	1	1	0	0
0	1	1	0	1	1	1	0	1
0	1	1	1	0	1	1	1	0
0	1	1	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	1	1	0
1	0	0	1	0	1	1	0	1
1	0	0	1	1	1	1	0	0
1	0	1	0	0	1	0	1	1
1	0	1	0	1	1	0	1	0
1	0	1	1	0	1	0	0	1
1	0	1	1	1	1	0	0	0

Tabela A.13 (parte)

ENDEREÇAMENTO	**DADOS** (Valores digitalizados)
A_4 A_3 A_2 A_1 A_0	D_3 D_2 D_1 D_0
1 1 0 0 0	0 1 1 1
1 1 0 0 1	0 1 1 0
1 1 0 1 0	0 1 0 1
1 1 0 1 1	0 1 0 0
1 1 1 0 0	0 0 1 1
1 1 1 0 1	0 0 1 0
1 1 1 1 0	0 0 0 1
1 1 1 1 1	0 0 0 0

Tabela A.13

fclock = $1MH_Z$

8.7.17

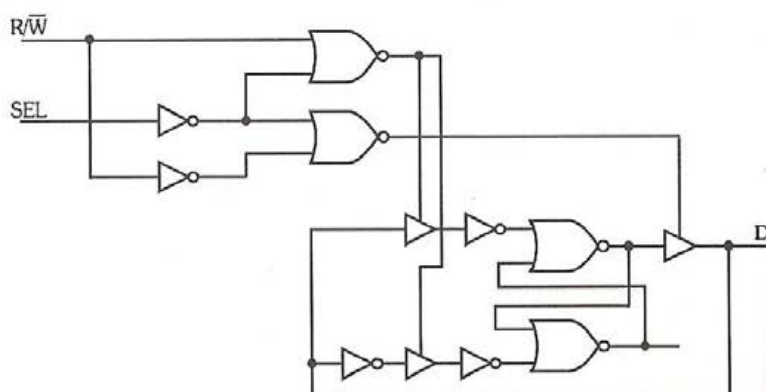



Figura A.69

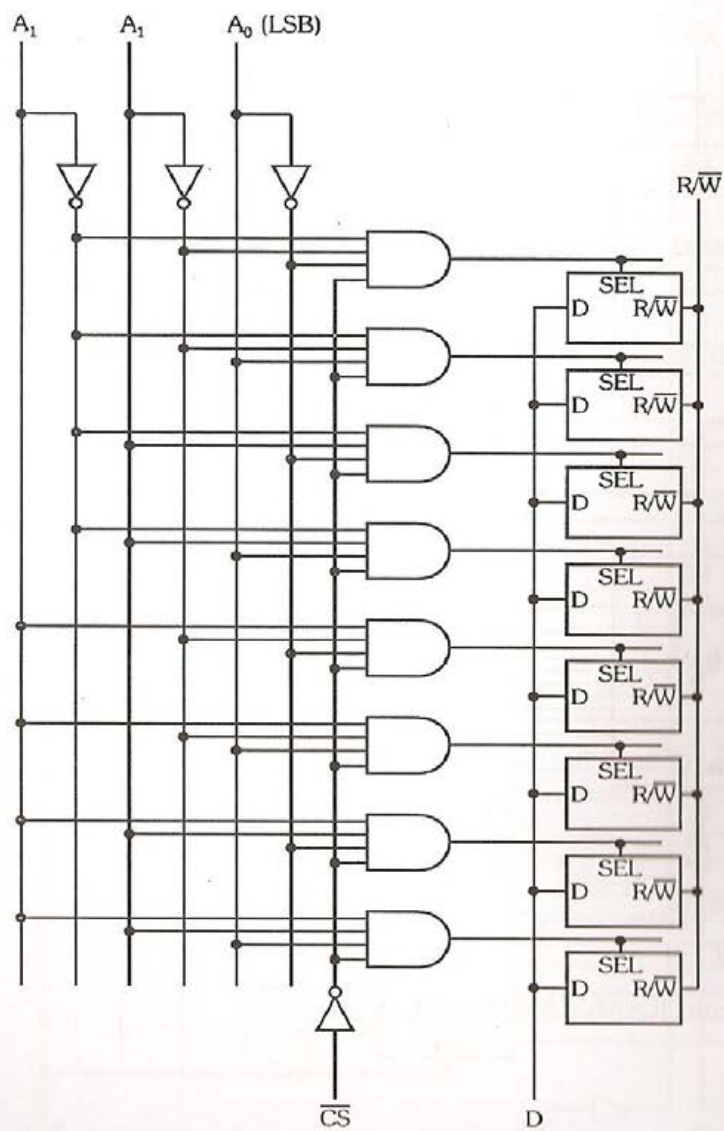



Figura A.70

8.7.19

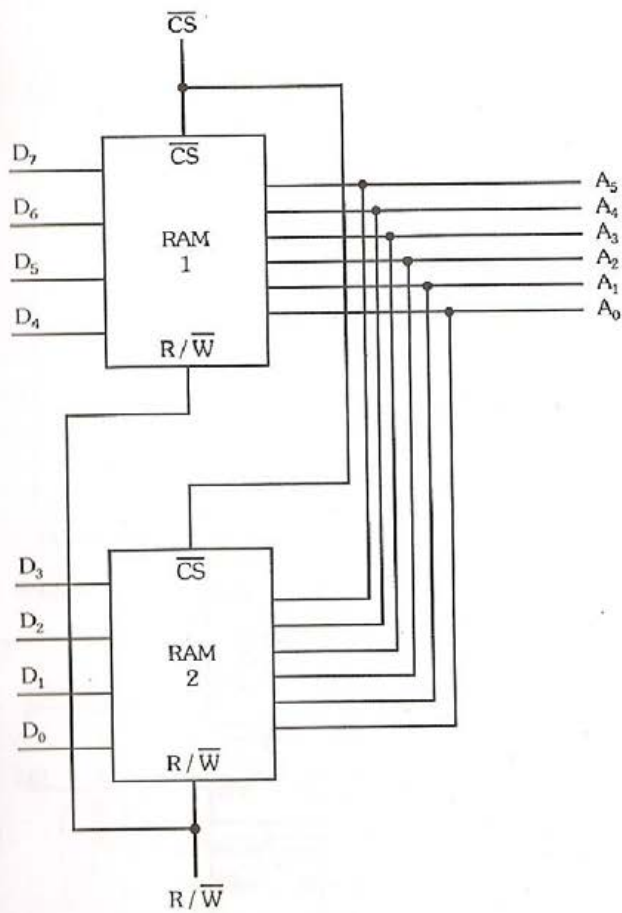



Figura A.71

Endereçamento RAM_1 e RAM_2 : 00_{16} a $3F_{16}$

8.7.20

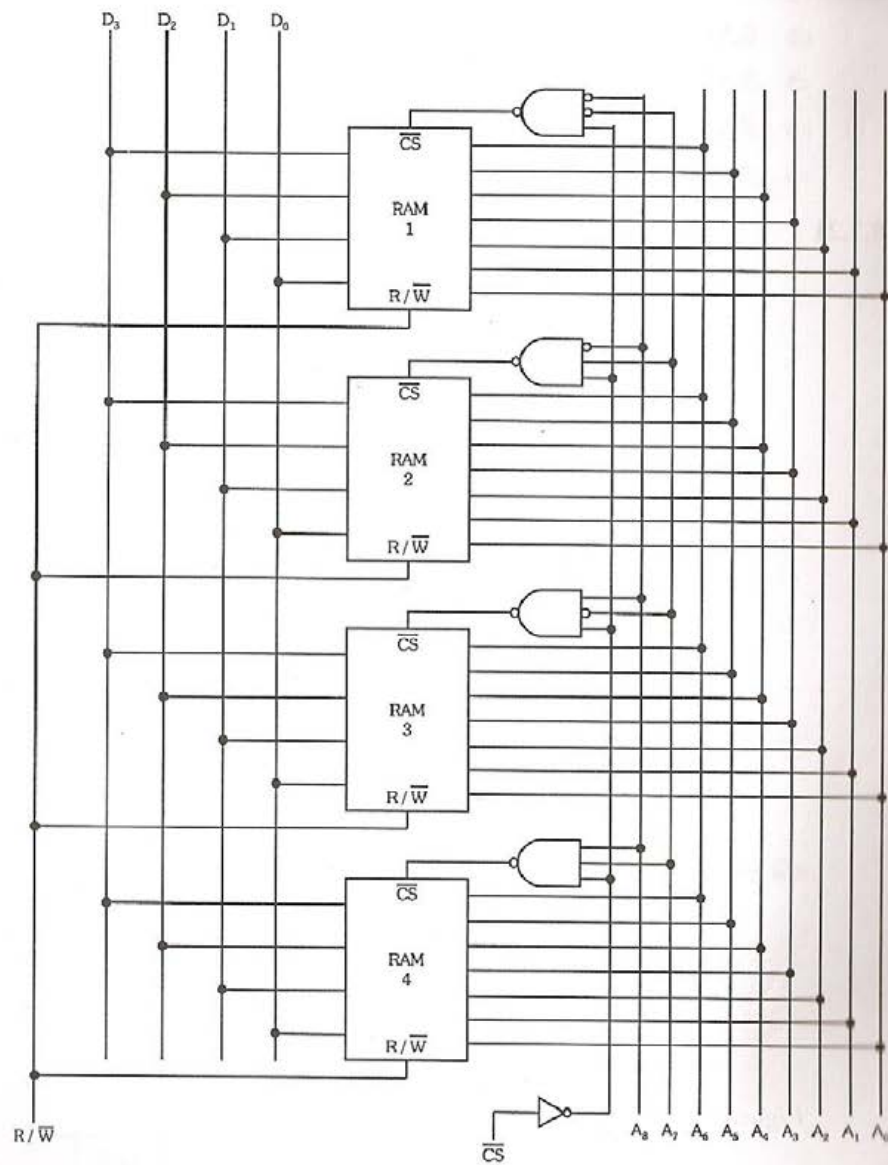



Figura A.72

Endereços:

⇨ RAM_1: 000_{16} a $07F_{16}$

⇨ RAM_2: 080_{16} a $0FF_{16}$

⇨ RAM_3 : 100_{16} a $17F_{16}$

⇨ RAM_4 : 180_{16} a $1FF_{16}$

8.7.21

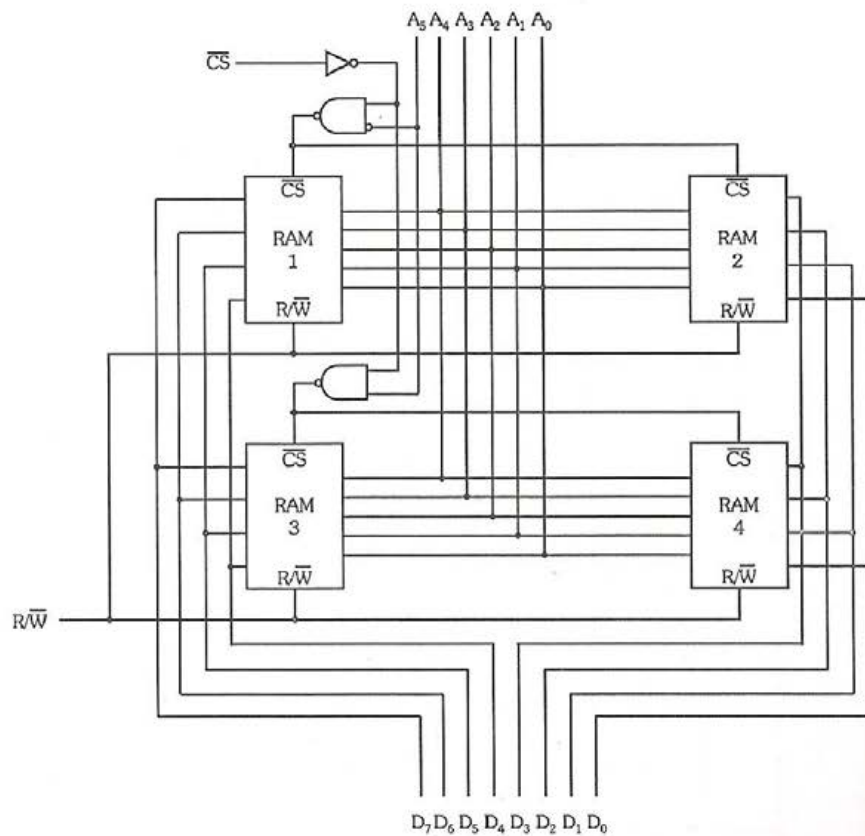



Figura A.73

Endereços:

⇨ RAM_1 e RAM_2 : 00_{16} a $1F_{16}$

⇨ RAM_3 e RAM_4 : 20_{16} a $3F_{16}$

Capítulo 9 - Famílias de Circuitos Lógicos

9.8.1

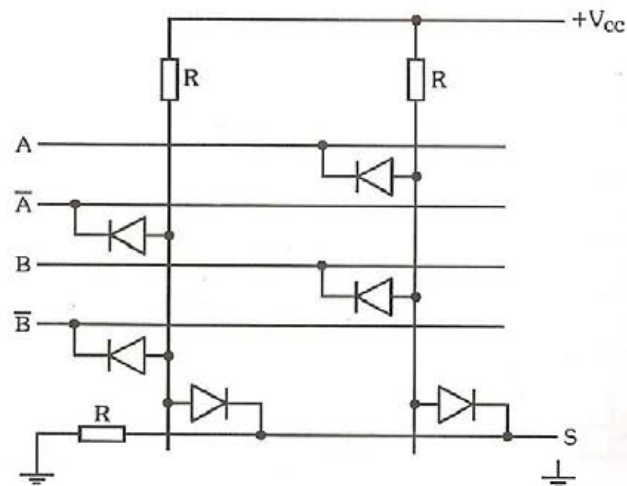



Figura A.74

9.8.2

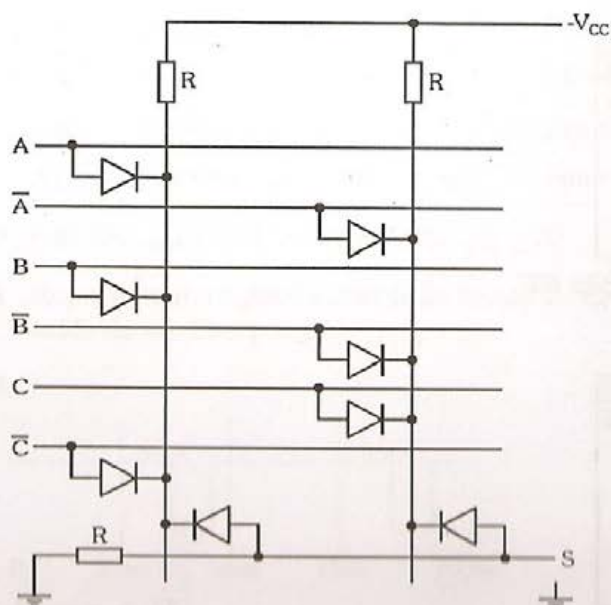



Figura A.75

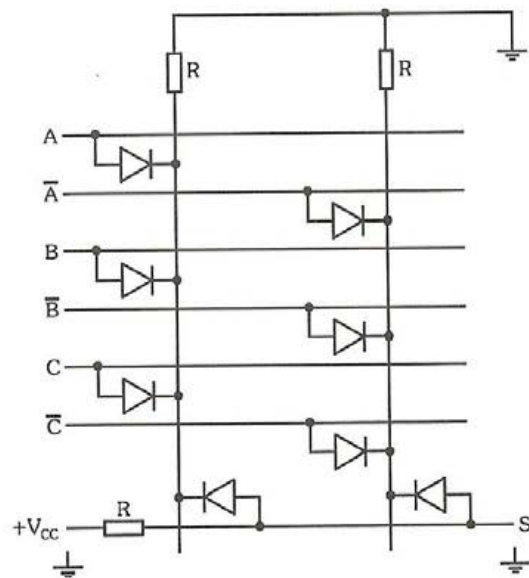



Figura A.76

9.8.4

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela A.14 Função NE

9.8.5

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

Tabela A.15 Função OU

9.8.6

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

Tabela A.16 Função NE

9.8.7 Aumentar o potencial da base do transistor, evitando que se iguale a potencial do diodo aterrado (nível 0).

9.8.8

A	M_1	M_2	S
0	conduz	corta	1
1	corta	conduz	0

Tabela A.17

9.8.9 S: $\text{Fan-Out}_{(\text{nível } 0)} = 10$ e $\text{Fan-Out}_{(\text{nível } 1)} = 20$

AS: $\text{Fan-Out}_{(\text{nível } 0)} = 40$ e $\text{Fan-Out}_{(\text{nível } 1)} = 100$

LS: $\text{Fan-Out}_{(\text{nível } 0)} = 20$ e $\text{Fan-Out}_{(\text{nível } 1)} = 20$

ALS: $\text{Fan-Out}_{(\text{nível } 0)} = 80$ e $\text{Fan-Out}_{(\text{nível } 1)} = 20$

9.8.10 $\text{Fan-Out}_{(\text{nível } 0)} = 40$ e $\text{Fan-Out}_{(\text{nível } 1)} = 20$

9.8.11 Representam respectivamente as faixas de nível indefinido de entrada saída de um bloco lógico.

9.8.12

Figura A.77

9.8.13

f = 50 Hz

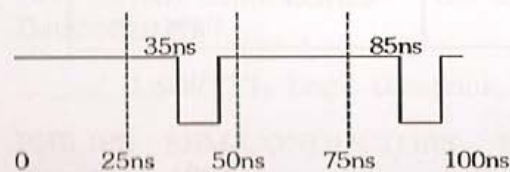
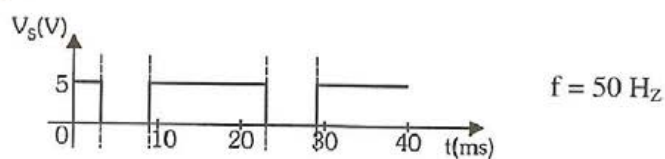



Figura A.78

9.8.14

FAMÍLIA	Tensão de Alimentação	Potência Dissipada	Margem de Imunidade ao Ruído	Tempo de Atraso de Propagação
TTL Standard	5V	10mW	0,4V	10ns
CMOS 4000B	3 a 15V	1nW	45%V_{DD}	90ns
comentários	Larga faixa em CMOS	Muito baixa em CMOS	Alta em CMOS	Baixo em TTL

FAMÍLIA	Velocidade	Fan-Out	Manuseio
TTL Standard	Alta	10	Livre
CMOS 4000B	Baixa	50	Com problemas
comentários	Alta em TTL	Alto em CMOS	Problemas em CMOS

BIBLIOGRAFIA

ADVANCED MICRO DEVICES. **AMD: Memory Products.** D Book, 1990.

CAEIRO, F. J. **Trabucho: Introdução à Eletrônica Digital.** vol Livros Horizonte, 1986.

CAPUANO, F. G. **Exercícios de Eletrônica Digital.** São Paulo: Éri 1996.

COPPELLI, M.; STORTONI, B. **Esercizi di Elettronica Digita** Fermo: La Sovrana Editrice, 1984.

DEBOO, G. J.; BURROUS, C. N. **Circuitos Integrados Y Dispositiv Semiconductores.** Marcombo Boixareu Editores, 1977.

ENDERLEIN, R. **Microeletrônica: Uma Introdução ao Universo d Microchips, seu Funcionamento, Fabricação e Aplicações.** São Paul Editora da Universidade de São Paulo, 1994.

HILL, F. J.; PETERSON, G. R. **Introduction to Switching Theory a Logical Design.** John Wiley & Sons, 1981.

MALVINO, A. P. **Microcomputadores e Microprocessadore** Mc GrawHill, 1985.

MALVINO, A. P.; LEACH, D. P. **Eletrônica Digital, Princípios Aplicações.** vols. 1 e 2. Mc GrawHill, 1988.

MELO, M. **Eletrônica Digital.** Mc GrawHill, 1993.

MOTOROLA. **Memory Data**, 1990.

NATIONAL SEMICONDUCTOR CORPORATION. **ALS/AS Logi** Databook, 1987.

______. **LS/S/TTL.** Logic Databook, 1987.

PHILIPS SEMICONDUCTORS. **Fast TTL Logic Series.** Dat Handbook, 1992.